Pontifícia Universidade Católica do Rio Grande do Sul
Faculdade de Informática
Programa de Pós-Graduação em Ciência da Computação

PROCESSO DE ETC ORIENTADO A SERVIÇOS PARA UM AMBIENTE DE GESTÃO DE PDS BASEADO EM MÉTRICAS

Patrícia Souza Silveira

Dissertação apresentada como requisito parcial à obtenção do grau de mestre em Ciência da Computação

Orientador: Prof. Dr. Duncan Dubugras Alcoba Ruiz

Porto Alegre
2007

Dados Internacionais de Catalogação na Publicação (CIP)

S587p Silveira, Patrícia Souza

Processo de ETC orientado a serviços para um ambiente de gestão de PDS baseado em métricas / Patrícia Souza Silveira. – Porto Alegre, 2007.

168 f.

Diss. (Mestrado em Ciência da Computação) – Fac. de Informática, PUCRS.
Orientação: Prof. Dr. Duncan Dubugras Alcoba Ruiz.

1. Informática. 2. Engenharia de Software.

3. Software – Técnicas de Avaliação. 4. Qualidade em Informática. 5. Data Warehousing. I. Ruiz, Duncan Dubugras Alcoba.

CDD 005.1

Ficha Catalográfica elaborada pelo
Setor de Tratamento da Informação da BC-PUCRS

Pontifícia Universidade Católica do Rio Grande do Sul
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TERMO DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO

Dissertação intitulada "**Processo de ETC Orientado a Serviços para um Ambiente de Gestão de PDS Baseado em Métricas**", apresentada por Patrícia Souza Silveira, como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação, Sistemas de Informação, aprovada em 21/12/07 pela Comissão Examinadora:

Prof. Dr. Duncan Dubugras Alcoba Ruiz – Orientador — PPGCC/PUCRS

Prof. Dr. Jorge Luis Nicolas Audy – — PPGCC/PUCRS

Profa. Dra. Renata de Matos Galante – — UFRGS

Homologada em 19/03/2008, conforme Ata No. 05/2008 pela Comissão Coordenadora.

Prof. Dr. Fernando Gehm Moraes
Coordenador.

PUCRS
Campus Central
Av. Ipiranga, 6681 – P32 – sala 507 – CEP: 90619-900
Fone: (51) 3320-3611 – Fax (51) 3320-3621
E-mail: ppgcc@inf.pucrs.br
www.pucrs.br/facin/pos

AGRADECIMENTOS

Agradeço a minha família, João, Jussara, Leonardo, Mico, Assis, Ostília, Altair, Leda, Guilherme, Leandro, Tânia e Coutelle, por todo apoio e carinho durante os momentos complicados desta etapa da minha formação acadêmica. Em especial a minha mãe que nas horas mais difícies conseguiu transmitir a força e o carinho durante longas conversas por telefone.

Ao Luiz, pelo amor, incentivo, dedicação e, principalmente, paciência nos meses finais de conclusão deste trabalho. Tua companhia foi fundamental.

A Raquel, minha segunda mãe, por todo o carinho e pelas horas de conversa pelo MSN, tua companhia virtual também foi primordial nos momentos mais díficieis

As minhas novas amigas Paty, Fernandinha e Aninha, que conviveram comigo nesses dois últimos anos e acompanharam de perto esta conquista, escutando minhas ansiedades e sempre oferencedo apoio.

Aos ilustres membros Gurizada-RG, Alex ,Bruno, Clayton, Daniel, de Bem, Gonças, Mariana, Maurano, Ralf, Rogérioe Velloso, pelos churrascos, formaturas, casamentos, emails engraçados, festas e Laruchos.

Aos meus amigos Gera, Ana, Gui, Sapo, Gordo,Cony, Bicho, Vanessa, Lidi, Angélica, Sônia e Velloso, pelo apoio e carinho.

Aos companheiros de mestrado Primo, Borges, Silva, Josi, Chinelo, Fabrício, Hugo, Bogoni, Márcio, Rafael e Cristian.

Ao meu orientador, professor Duncan Ruiz, pelos, ensinamentos, conselhos e incentivo durante a conclusão deste trabalho.

Ao Convênio HP EAS Brasil/PUCRS por viabilizarem a bolsa de estudos bem como todo o aprendizado fornecido durante esse tempo, principamente a Taísa e o Fidelix.

Ao grupo GPIN (Grupo de Pesquisa em Inteligência de Negócio) pelo conhecimento trocado, apoio em diversos momentos, que provaram que na pesquisa se faz grandes amigos.

Ao Programa de Pós-Graduação em Ciência da Computação e a todos os professores dos quais pude conviver durante esse período.

E as demais pessoas que durante esse tempo colaboraram de alguma forma com esse trabalho.

“Quando a lua apareceu ninguém sonhava mais do que eu
Já era tarde, mas a noite é uma criança distraída
Depois que eu envelhecer ninguém precisa mais me dizer
Como é estranho ser humano nessas horas de partida
É o fim da picada, depois da estrada começa uma grande avenida
No fim da avenida, existe uma chance, uma sorte, uma nova saída
São coisas da vida”

Rita Lee

RESUMO

A busca pela qualidade é uma constante nos ambientes corporativos. Para tanto, as operações de desenvolvimento de *software* utilizam métricas para mensurar a qualidade dos seus produtos, processos e serviços. As mesmas devem ser coletadas, consolidadas e armazenadas em um repositório central único, tipicamente implementado na forma de *Data Warehouse* (*DW*). A definição do processo de extração, transformação e carga (ETC) das métricas a serem armazenadas nesse repositório, considerando as características do ambiente de desenvolvimento de *software* (heterogeneidade de fontes, de modelos de processos, de tipos de projetos e de níveis de isolamento) não é uma tarefa trivial.

Este trabalho apresenta um ambiente de *data warehousing* denominado *SPDW+*, como solução para a automatização do processo de ETC das métricas. Esta solução contém um modelo analítico abrangente e elegante, para análise e monitoração de métricas, e é baseada em uma abordagem orientada a serviços, aliada à tecnologia de *Web Services* (*WS*). Além disso, o *SPDW+* trata a carga incremental com baixo nível de intrusão, e alta freqüência e baixa latência na coleta das métricas. Os principais componentes da solução são especificados, implementados e testados.

Os benefícios desta solução são: i) ser flexível e adaptável para atender às constantes modificações do ambiente do negócio; ii) oferecer suporte à monitoração, permitindo a realização de cargas freqüentes e incrementais; iii) ser capaz de desonerar os projetos da tarefa, laboriosa e complexa, de captura das métricas; iv) manter a liberdade de escolha dos projetos, quanto aos modelos de gestão e às ferramentas de apoio empregadas; e v) possibilitar que as informações contidas no repositório de métricas estejam coesas e consistentes, para que os dados de diferentes projetos sejam comparáveis entre si.

Palavras Chave: Métricas de *Software*, *Data Warehousing*, ETC, *Web Services* e AOS.

ABSTRACT

The search for quality is a constant value in corporate environments. With this aim, software development organizations utilize metrics to measure quality of their products, processes and services. These metrics should be collected, consolidated and stored in a single central repository typically implemented as a Data Warehouse (DW). The definition of extraction, transformation and loading (ETL) of metrics that will be stored in DW, considering the software development environment (heterogeneity of sources, of process models, of project classes and of level of isolation) is no trivial task.

This paper presents a data warehousing environment called SPDW+ as a solution to the automation of the ETL metrics process. This solution introduces a comprehensive and streamlined analytical model for the analysis and monitoring of metrics, and is built on a service-oriented approach that utilizes the Web Services technology (WS). Moreover, SPDW+ addresses the low-intrusion incremental load and the high frequency and low latency present in metrics collection. The main components of SPDW+ are specified, implemented and tested.

The advantages of SPDW+ are: (i) flexibility and adaptation to meet the requirements of the constant changes in business environments; (ii) support to monitoring, which allows the run of frequent and incremental loads; (iii) the capacity to make less burdensome the complex, time-consuming task of capturing metrics; (iv) freedom of choice regarding management models and the support tools used in projects; and (v) cohesion and consistency of the information contained in the metrics repository needed to compare the data of different projects.

Keywords: Software Metrics; Data Warehousing; ETL, Web Services and SOA.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

%TC	Percentual de Trabalho Completado
AOS	Arquitetura Orientada a Serviços
APM	*Abstract Process Monitor*
BI	*Business Intelligence*
BO	Base Organizacional
BPI	*Busines Process Intelligence*
BPMS	*Business Process Management System*
CBO	Custo *Baseline* Original
CBR	Custo *Baseline* Revisado
CMM	*Capability Maturity Model*
CMMI	*Capability Maturity Model Integration*
CORBA	*Common Object Request Broker Architecture*
CR	Custo Real
CRAR	Custo Real da Atividade de Revisão
CRAQ	Custo Real da Atividade de Qualidade
CRFT	Custo Real da Fase de Teste
DCOM	*Distributed component object model*
DDE	Densidade de Defeitos Externos
DDI	Densidade de Defeitos Internos
DE	Defeitos Externos
DFBO	Data Final *Baseline* Original
DFBR	Data Final *Baseline* Revisado
DFR	Data Final Real do Projeto
DI	Número defeitos encontrados internamente
DIBO	Data Inicial *Baseline* Original
DIBR	Data Inicial *Baseline* Revisado
DS	Data *Status*
DSA	*Data Staging Area*
DTS	*Data Transformation Services*
DW	*Data Warehouse*
EBO	Esforço *Baseline* Original

EBR	Esforço *Baseline* Revisado
EPM	*Enterprise Project Manager*
ER	Esforço Real
ERD	Eficiência de Remoção de Defeitos
ERV	Eficiência de Revisão
ETC	Extração, Transformação e Carga
EVA	*Earned Value Analysis*
GPIN	Grupo de Pesquisa em Inteligência de Negócio
HP	*Hewlett-Packard*
HTTP	*Hypertext Transfer Protocol*
iBOM	*Intelligence Operations Manager*
IDC	Índice de Desempenho de Custo
IDP	Índice de Desempenho de Prazo
IEEE	*Institute of Electrical and Electronics Engineers*
IIS	*Internet Information Service*
ISC	Índice de Satisfação do Cliente
ISO	*International Organization for Standardization*
KLOC	*Kilo Line of Codes*
NDI	Número de Defeitos Internos
NDE	Número de Defeitos Externos
NMA	Número de Modificações Aprovadas
NMR	Número de Modificações Requeridas
MA	*Measurement and Analysis*
MDX	*Multidimensional Expressions*
MMR	*Multidimensional Measurement Repository*
MRE	Número de Requisitos Excluídos
MRM	Número de Requisitos Modificados
MS	*Microsoft*
OLAP	*On-Line Analytical Process*
OSSP	*Organization's Set of Standard Process*
PA	*Process Area*
PDS	Processo de Desenvolvimento de *Software*
PDSP	*Project's Defined Software Process*

PDP	*Project's Defined Process*
PF	Ponto de Função
PMHP	Ponto Médio HP
PR	Produtividade
RMI	*Remote Method Invocation*
SC	Satisfação do Cliente
SDP	*Software Development Process*
SOAP	*Simple Object Access Protocol*
SSIS	*SQL Server Integration Service*
SPICE	*Software Process Improvement and Capability Determination*
SPDW	*SDP Performance Data Warehousing*
SPDW+	*SDP Performance Data Warehousing Plus*
SQL	*Structured Query Language,*
TBO	Tamanho *Baseline* Original
TBR	Tamanho *Baseline* Revisado
TI	Tecnologia de Informação
TR	Tamanho Real
UDDI	*Universal Description, Discovery and Integration*
UML	*Unified Modeling Language*
VA	Valor Agregado
VBO	Variação do *Baseline* Original
VBR	Variação do *Baseline* Revisado
VC	Variação de Custo
VCA	Variação de Custo Agregada
VE	Valor Estimado
VEBO	Variação de Esforço do *Baseline* Original
VEBR	Variação de Esforço do *Baseline* Revisado
VP	Variação de Prazo
VPA	Variação de Prazo Agregada
VR	Volatilidade de Requisitos
W3C	*World Wide Web Consortium*
WS	*Web Services*
WSDL	*Web Services Description Language*

VTBO	Variação de Tamanho do *Baseline* Original
VTBR	Variação de Tamanho do *Baseline* Revisado
XML	*Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

A busca pela qualidade é uma constante nos ambientes cooporativos. Para tanto, as operações de desenvolvimento de *software* utilizam métricas para mensurar a qualidade dos seus produtos, processos e serviços. As mesmas devem ser coletadas, consolidadas e armazenadas de maneira consistente, pouco onerosa e estruturada em um repositório central único, a partir do qual seja possível oferecer suporte à tomada de decisão e, dessa forma, permitir a análise, monitoração e previsão das métricas de qualidade [SEI06]. Segundo [INM05], *Data Warehouse* (*DW*) é o cerne para a construção de sistemas de apoio à decisão. A definição do processo de extração, transformação e carga (ETC) dos dados a serem armazenados em um *DW* representa uma das tarefas mais dispendiosas durante o estabelecimento de um ambiente de *data warehousing* [KIM98], podendo consumir pelo menos um terço do esforço e do orçamento inicialmente estimado para a construção do mesmo.

A heterogoneidade e as particularidades do ambiente de desenvolvimento de *software*, tornam a definição de um processo de ETC de métricas uma tarefa não trival [BEC06][CUN05]. Poucas propostas de repositório de dados que ofereçam suporte à ETC de métricas, de maneira automatizada, foram encontradas na literatura. Dentre elas destacam-se: *MMR* (*Multidimensional Measurement Repository*) [PAL03], *BPI* (*Busines Process Intelligence*) [CAS02b], *iBOM* (*Intelligence Operations Manager*) [CAS05] e *SPDW* (*SDP Performance Data Warehousing*) [BEC06]. No entanto, nenhuma delas contempla de maneira adequada o suporte à análise, monitoração e previsão da mensuração da qualidade de *software*, a partir de métricas, de maneira pouco intrusiva e considerando um ambiente dinâmico e heterogêneo de desenvolvimento de *software*.

Esta pesquisa visa facilitar a gestão de um ambiente de desenvolvimento de *sofware* ao propor a automatização do processo de ETC de métricas, a serem armazenadas em um repositório estruturado na forma de um *DW*, e utilizar uma abordagem orientada a serviços aliada à tecnologia de *Web Services* (*WS*). A solução desenvolvida, implementada e testada, denomina-se *SPDW+*[1] e é uma extensão substancial em relação à *SPDW* [BEC06] [RUI05]. Esta solução apresenta um modelo analítico abrangente e elegante, que oferece suporte à análise e monitoração segundo a técnica de *Earned Value Analysis* (*EVA*) [PMI04] para monitorar custos e prazos de projetos.

[1] Acrônimo pronunciado como "*speedway plus*".

A solução proposta apresenta como benefícios: i) ser flexível e adaptável para atender às constantes modificações do ambiente do negócio; ii) oferecer suporte à monitoração, permitindo a realização de cargas freqüentes e incrementais; iii) ser capaz de desonerar os projetos da tarefa, laboriosa e complexa, de captura das métricas; iv) manter a liberdade de escolha dos projetos, quanto aos modelos de gestão e às ferramentas de apoio; e v) possibilitar que as informações contidas no repositório de métricas sejam coesas e consistentes, para que os dados dos diferentes projetos sejam comparáveis entre si.

Os capítulos deste trabalho estão organizados da seguinte forma. O Capítulo 2 apresenta o referencial teórico essencial para o entendimento dos conceitos, técnicas e métodos adotados por esta pesquisa. O Capítulo 3 descreve o cenário de desenvolvimento, sob a perspectiva de três dimensões ortogonais distintas, que envolvem os aspectos relacionados com a automação da mensuração da qualidade de *software*, tabulados em um instrumento de refência. O Capítulo 4 relata um estudo de caso de uma operação de *software*, de grande porte, certificada *CMM3*. O Capítulo 5 descreve a solução proposta, denominada *SPDW+*, e a contribuição deste trabalho de pesquisa. O Capítulo 6 apresenta os testes realizados para avaliar se o processo automatizado de ETC do *SPDW+* consegue tratar adequadamente, e em tempo compatível, os aspectos que envolvem a mensuração da qualidade de *software*, segundo o instrumento de referência. O Capítulo 7 discorre sobre as considerações finais e os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem o intuito de permitir um melhor entendimento de alguns conceitos e tópicos relacionados com o tema pesquisa. Para tanto, o mesmo apresenta uma breve descrição sobre: (i) PDS e modelos de desenvolvimento de *software*; (ii) mensuração de PDS por intermédio de métricas de *software*, bem como a documentação das mesmas por intermédio de um Programa de Métricas organizacional; (iii) utilização da técnica de *EVA* para monitorar prazos e custos durante a execução de um projeto; (iv) ambientes de *data warehousing*, incluindo as técnicas e os passos utilizados para a sua construção; e (iv) padrões de *WS* envolvidos para desenvolver uma AOS (Arquitetura Orientada a Serviços).

2.1 Processo de Desenvolvimento de *Software* (PDS)

A disseminação da utilização de soluções computacionais nos mais variados cenários faz com que as organizações de Tecnologia de Informação (TI) busquem uma forma de produzir *software*, com o máximo de lucro, qualidade e satisfação do cliente. Dentro desse contexto, o desenvolvimento de *software* deixou de ser uma arte. O mesmo tornou-se um processo estruturado, organizado e bem definido, seguindo determinados padrões e conceitos, sendo muitas vezes uma tarefa complexa, envolvendo diversas pessoas e resultando em altos custos.

2.1.1 Modelos de Processo de *Software*

Para endereçar de maneira adequada as questões relacionadas com o desenvolvimento de *software*, as organizações de TI têm adotado estratégias de desenvolvimento, baseadas em processos, métodos e ferramentas. Essas estratégias são conhecidas como modelos de processo de *software*, também chamados de paradigmas de engenharia de *software*. Esses modelos devem ser escolhidos conforme a base e a natureza do projeto e da aplicação, bem como os métodos e as ferramentas utilizadas durante a concepção dos produtos [PRE04]. Por essas razões, a Engenharia de *Software* propõe diversos modelos de desenvolvimento, estabelecidos com base nas diferentes perspectivas e necessidades do PDS. É importante ressaltar que essa variedade de modelos de desenvolvimento de *software* é conseqüente da realidade distinta dos clientes e dos usuários dos sistemas. A seguir é apresentada uma breve

descrição de alguns modelos de processo de *software.* Maiores detalhes sobre esse assunto podem ser encontrados em [PRE04] [SOM04].

- **Modelo Cascata ou Seqüencial Linear**: divide a missão complexa de desenvolvimento de *software* em vários passos lógicos (projeto, codificação, teste, entre outros), com entregáveis intermediários, que conduzem ao produto final. Para assegurar a qualidade desses entregáveis cada passo possui um critério de validação, parâmetros de entrada e de saída.
- **Modelo Espiral**: cada porção do produto e nível de elaboração deve envolver a mesma seqüência de passos (ciclos), começando no centro da espiral, sendo que, cada fase de desenvolvimento compreende apenas um único ciclo da espiral. É tipicamente representado por duas dimensões, onde a dimensão radial representa o custo cumulativo para completar um determinado passo, enquanto que a dimensão angular caracteriza o progresso realizado obtido a partir da completude de cada ciclo do espiral.
- **Modelo Evolutivo ou Iterativo:** inicia com um subconjunto de requisitos para desenvolver um subconjunto de produtos, que satisfaz as necessidades essenciais de um determinado cliente. Esse modelo tem por objetivo proporcionar um veículo para análises e treinamento para o cliente, além de oferecer experiência de aprendizagem ao desenvolvedor. Com base nas análises recorrentes de cada produto intermediário, o projeto e os seus requisitos são modificados em uma série de iterações. Esse modelo combina a prototipação com a rigidez clássica do modelo cascata. Além disso, o mesmo tem muito em comum com o modelo espiral, principalmente com relação à gerência de riscos e a prototipação.
- **Processo Unificado:** consiste na repetição de uma série de ciclos durante o tempo de vida de desenvolvimento de um sistema. Cada ciclo concluído é considerado uma versão do produto pronta para distribuição. As versões devem apresentar um conjunto relativamente completo e consistente de artefatos, possivelmente incluindo manuais e um módulo executável do sistema, podendo ser distribuídas aos usuários internos e externos. Esses ciclos são formados por quatro fases distintas, denominadas: início, elaboração, construção e transição, sendo que cada uma delas pode ser subdivida em iterações.

Seja qual for o modelo de desenvolvimento de *software* adotado para um projeto, a realização do mesmo sempre pode ser planificada em função do tempo e estruturada segundo uma hierarquia de tarefas. Assim, em linhas gerais, pode-se dizer que um determinado projeto

apresenta versões, compostas de iterações que, por sua vez, são constituídas por fases e atividades. Dessa forma, quando todas as atividades que compõem um determinado nível hierárquico forem concluídas, o mesmo é considerado finalizado e assim, sucessivamente, até que o projeto seja finalizado por completo.

2.2 Mensuração do PDS

Segundo [KAN03], a atividade de mensuração é crucial para o avanço de qualquer ciência. O progresso científico é realizado a partir de observações e generalizações baseadas em dados e medições, e na derivação de teorias como resultados, bem como na confirmação ou refutação dessas teorias, por intermédio de testes de hipóteses, baseados em dados empíricos avançados. Ainda, conforme o mesmo autor, a qualidade deve ser definida operacionalmente, mensurada, monitorada, gerenciada e melhorada.

No contexto de desenvolvimento de *software* a mensuração da qualidade, dos processos, produtos e projetos, é determinada por métricas. Segundo os modelos de qualidade [SEI06], é necessário realizar a mensuração das características e dos parâmetros de qualidade do PDS nos seus diferentes estágios, bem como estabelecer métricas e modelos para assegurar que o processo de desenvolvimento esteja sob controle e no caminho do cumprimento dos objetivos de qualidade do produto.

O desafio de capturar métricas de *software* está em ter a certeza que a coleta proporciona informação útil à gestão de qualidade e que essa captura é realizada de uma forma pouco intrusiva para o time de desenvolvimento [PRE04]. Intrusão neste contexto significa deslocar recursos do projeto (colaboradores), de suas atividades normais, para realizar a coleta de métricas. Coletar dados de Engenharia de *Software* pode ser dispendioso. No entanto, coletar dados e analisá-los produz inteligência e conhecimento sobre os projetos e o seu PDS, além de ser vital para o sucesso do negócio.

2.2.1 Métricas de *Software*

Uma métrica [IEE98] é uma função mensurável, cujas entradas são dados de *software* e seu resultado corresponde a um único valor numérico, que pode ser interpretado como o grau de qualidade do *software*, sendo esse afetado por um determinado atributo. Esse último pode ser definido como “uma propriedade física ou abstrata mensurável de uma entidade”.

Um fator de qualidade é um tipo de atributo orientado à gestão de *software*, que contribui para sua qualidade.

Ainda conforme a *IEEE* [IEE98], as métricas de *software* são classificadas como diretas e derivadas. As diretas não dependem de nenhum outro atributo, nem necessitam de validação, enquanto que as derivadas são definidas através de outros atributos e precisam ser validadas. Ambas têm por objetivo proporcionar uma informação quantitativa sobre um processo ou um produto, sendo importantes às atividades de gerenciamento de *software* e ao controle de desempenho de processos, tanto no âmbito da organização, como no ambiente de cada projeto.

Segundo [KAN03], as métricas de *software* também podem ser classificadas em três categorias: produto, processo e projeto. A primeira descreve as características do produto (*e.g.* tamanho, complexidade, peculiaridades do projeto, desempenho e nível de qualidade). A segunda pode ser utilizada para melhorar o desenvolvimento e a manutenção do PDS (*e.g.* defeitos encontrados e o esforço despendido para a sua correção, eficiência de remoção de defeitos). A terceira representa as características do projeto e da sua execução (*e.g.* número de recursos, custo, cronograma e produtividade).

Ainda, conforme o mesmo autor, as métricas podem apresentar domínios de valores em uma hierarquia de quatro escalas, descritas abaixo na forma *bottom-up*:

- **Escala nominal:** a classificação é considerada a operação mais simples da ciência e representa o nível mais baixo de mensuração. Classificar consiste na tarefa de separar elementos de acordo com um determinado atributo. Por exemplo, com base nos seus ciclos de vida pode-se classificar os modelos de desenvolvimento de s*oftware* em: cascata, evolutivo, espiral, entre outros. Dessa forma, a escala nominal é composta por categorias, que precisam ser mutuamente exclusivas e, quando agrupadas, devem cobrir todas as possibilidades de categorias de um atributo. Quando essas duas exigências são atingidas, estão estabelecidas as condições necessárias para a realização de análises estatísticas. Nesse sentido, podem ser realizadas comparações tendo como ponto de partida atributos de interesse como: taxa de defeito, intervalo de tempo e defeitos, dentre as diferentes categorias de produtos de *software*.

- **Escala ordinal:** refere-se a operações de mensuração em que as medidas podem ser comparadas de maneira ordenada. Por exemplo, os processos de um determinado projeto podem ser mensurados conforme seu grau de aderência aos modelos de maturidade em: totalmente aderente, parcialmente ou não aderente. Segundo a

hierarquia de mensuração, a escala ordinal representa um nível maior que a escala nominal. A partir da primeira é possível agrupar as métricas em categorias e estabelecer uma ordem de comparação. A escala ordinal é considerada antissimétrica e transitiva, logo (A > B) é verdadeiro, porém (B > A) é falso. Além disso, se (A > B) e (B > C) então (A > C). Assim, com base nessas propriedades, é possível oferecer informações sobre a magnitude da diferença entre elementos distintos.

- **Escala intervalar e de razão:** escala intervalar indica a diferença exata entre dois pontos de medição, aceita operações matemáticas de adição e subtração e deve apresentar uma unidade de medida bem definida. Já a escala de razão, é obtida quando um zero absoluto pode ser alocado na escala intervalar. Assim, essa última representa o nível mais alto de mensuração, e permite a aplicação de todas as operações matemáticas, incluindo divisões e multiplicações. Em ambas, os valores podem ser expressos, tanto em inteiros, como em reais.

Essas escalas são consideradas hierárquicas, pois os níveis mais altos apresentam as propriedades dos níveis inferiores. O nível mais simples e limitado é definido pela escala nominal, permitindo somente a detecção de categorias. Em seguida, tem-se a escala ordinal, que possibilita diferenciar patamares. Posteriormente, tem-se a escala intervalar, que proporciona o posicionamento de valores em relação a um ponto arbitrário. Finalmente, a mais importante de todas é a escala de razão, que admite a comparação de valores em termos absolutos. Assim, quanto mais alto o nível da mensuração, melhores são as análises que podem ser aplicadas.

2.2.1.1 Programa de Métricas

Para permitir a institucionalização e fomentar o uso das métricas de *software*, as mesmas devem ser documentadas e organizadas por intermédio de um Programa de Métricas [SEI06]. Esse deve conter a definição das métricas adotadas pela organização, as suas respectivas unidades de medida, bem como as equações utilizadas para o seu cálculo, no caso das métricas derivadas [KAN04]. Tanto a norma 1061 [IEE98], como os modelos de qualidade (*CMM* (*Capability Maturity Model*) e *CMMI (Capability Maturity Model Integration)*), não definem um conjunto mínimo de métricas. Ambos apenas afirmam que devem ser estabelecidas métricas úteis e significativas, conforme as necessidades dos projetos e da organização. Nesse sentido, o principal desafio durante a definição de um Programa de

Métricas está intrinsecamente ligado com a identificação da relação entre os objetivos da organização e os objetivos das métricas ([OTL01] *apud* [HAY03]), buscando proporcionar informações adequadas aos diferentes níveis organizacionais.

Segundo a norma 1061, cada métrica que pertencente ao Programa de Métricas organizacional deve apresentar: (i) o seu respectivo nome; (ii) o custo, os benefícios e os impactos associado à sua utilização; (iii) as faixas de valores esperados que correspondam ao desempenho das métricas; iv) as ferramentas utilizadas para armazenar, computar e avaliar os resultados; (v) a aplicabilidade da métrica, descrevendo como a mesma pode ser utilizada e quais as suas áreas; (vi) os valores de entrada necessários para o cálculo da métrica; e (vii) um exemplo de utilização para cada uma das métricas.

As métricas podem ser logicamente agrupadas em áreas de qualidade, segundo as expectativas e/ou necessidades de análise de uma determinada organização. A definição de um Programa de Métricas que contemple todas as áreas de qualidade desejadas por uma organização é de suma importância, já que a análise de uma métrica de forma isolada apresenta pouco valor. Avaliá-las dentro do contexto de um Programa de Métricas propicia acompanhar a história de um processo ou organização.

2.3 *Earned Value Analysis* (*EVA*)

O conceito de valor agregado teve sua origem no Departamento de Defesa Americano (DOD – *Department of Defense*), em 1967, com o intuito de controlar e administrar riscos e custos de projetos e programas [DEP07]. A análise do mesmo surgiu da necessidade de realizar previsões confiáveis relacionadas com custos e prazos, tendo como foco a relação entre o custo real e o produto físico produzido no projeto, por intermédio de uma quantidade fixa de trabalho e dentro de um prazo determinado.

O conceito de valor agregado exige que as medidas de despesa-produto sejam estabelecidas dentro de um cronograma físico de projeto. Assim, a partir da relação entre o valor agregado e o trabalho estimado no tempo, é possível realizar um controle do andamento das atividades de maneira mais precisa do que com método tradicional, que analisa esses fatores de forma isolada [FLE00].

A *EVA* permite tomar ações corretivas e preventivas com antecedência, pois oferece, aos gestores do projeto, uma forma de acompanhar o andamento das atividades para verificar se as mesmas estão sendo realizadas dentro do custo estimado, ou não, e se há algum tipo de

atraso ou aceleração da execução das atividades. Para tanto, são definidos alguns termos [PMI04]:

- **Valor Estimado (VE):** indica a parcela de orçamento que deveria ser gasta, segundo as estimativas realizadas no início do projeto (linha base), considerando as atividades, as atribuições ou os recursos envolvidos. O VE é definido como: o valor total do orçamento planejado, dividido pelo tempo total e acumulado até a data atual do projeto ou de *status*.

- **Valor Agregado (VA):** indica a parcela do orçamento que deveria ser gasta, considerando o trabalho realizado até o momento e o custo da linha base das atividades, das atribuições ou dos recursos. Esse valor é obtido por intermédio do produto entre o percentual de trabalho completado até a data de *status* e o valor total do orçamento estimado para determinada atividade.

- **Custo Real (CR):** é o custo total decorrido do trabalho já realizado, por um recurso ou atividade, até a data de *status* ou data atual do projeto. O valor do CR deve ser obtido levando em conta os mesmos dados usados no cálculo de VE e VA. Por exemplo: somente horas diretas, apenas custos diretos ou todos os custos, incluindo os indiretos.

A *EVA* é realizada a partir da relação entre esses termos. A Figura 1 contém uma representação gráfica de um possível conjunto de valores de VE, VA e CR. O gráfico apresenta como ordenada os valores de tempo e como abscissa o custo (valor monetário). As três curvas do gráfico representam os valores dos termos ao longo do tempo, até uma determinada data de *status*. As posições relativas entre as mesmas variam conforme os valores do projeto que está sendo analisado. Observa-se no gráfico que a referência para análise é a data de *status*. Cabe salientar que somente VE ultrapassa essa data, já que representa o valor estimado para toda a atividade ou projeto.

Para analisar a relação entre os valores de VE, VA e CR foram definidas algumas variações:

- **Variação de Custo (VC)**: é a diferença entre o valor agregado (VA) e o custo real (CR). Se a mesma for positiva, o custo está abaixo do valor estimado; caso contrário encontra-se acima do orçamento estipulado.

$$VC = VA - CR \qquad \begin{array}{l} VC \geq 0 \therefore \text{dentro do orçamento} \\ VC < 0 \therefore \text{fora do orçamento} \end{array}$$

- **Variação de Prazo (VP):** é a diferença entre o valor agregado (VA) e o valor estimado (VE). Se a mesma for positiva, o projeto está adiantado; caso o contrário é considerado atrasado.

$$VP = VA - VE$$

$VP > 0 \therefore$ adiantado
$VP = 0 \therefore$ no prazo
$VP < 0 \therefore$ atrasado

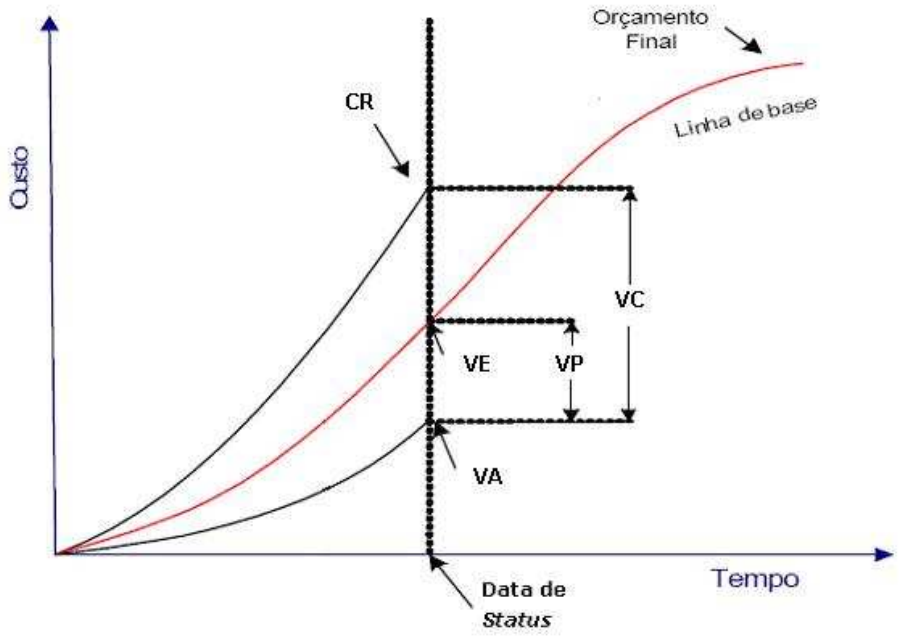


Figura 1 - Identificação das Variações no Gráfico de *EVA* [VAR04].

A Figura 1 ilustra como essas variações são identificadas no gráfico da *EVA*. A partir da mesma pode-se concluir que a variação de custo (VC) e de prazo (VP) pode ser considerada como a diferença entre as curvas VA e CR e as curvas VA e VE, respectivamente. Quanto mais distante a curva CR estiver das demais maior será a variação do prazo e no custo planejado para aquela data de *status*. Os valores VC e VP ainda podem ser convertidos em indicadores de eficiência para refletir o desempenho de custos e de prazos de qualquer projeto.

- **Índice de Desempenho de Custo (IDC):** relaciona o custo do valor agregado com o executado. Um valor de IDC menor que 1 indica que orçamento consumido foi menor que o planejado. Já um valor maior que 1 representa que os custos planejados não foram atingidos. O IDC é igual à razão entre VA e CR e mostra

qual a conversão entre os custos reais consumidos pelo projeto e os valores agregados em um determinado período. Por exemplo, um IDC = 0,83 indica que, para cada R$ 1,00 de capital realmente consumido, apenas R$ 0,83 estão sendo convertidos fisicamente em produto e que existe uma perda de $0,17.

$$IDC = \frac{VA}{CR} \qquad \begin{array}{l} IDC \geq 1 \therefore \text{ menor que o planejado} \\ IDC < 1 \therefore \text{ maior que o planejado} \end{array}$$

- **Índice de Desempenho de Prazo (IDP):** O IDP é usado, em adição ao andamento do cronograma, para prever a data de término, e representa a taxa de conversão do valor estimado (VE) em valor agregado (VA). O mesmo também pode ser utilizado em conjunto com o IDC para prever as estimativas de término do projeto. Por exemplo, um IDP = 0,85 indica que 85% do tempo estimado no orçamento (VE) foi convertido em trabalho e que houve uma perda de 15% no tempo disponível.

$$IDP = \frac{VA}{VE} \qquad \begin{array}{l} IDP \geq 1 \therefore \text{ dentro do prazo} \\ IDP =< 0 \therefore \text{ fora do prazo} \end{array}$$

2.4 Ambiente de *Data Warehousing*

Um *data mart* consiste em uma base de dados para armazenar um único processo de negócio ou um grupo de processos de negócio voltados para alcançar determinado objetivo. Nesse sentido, um *DW* é definido como a união de todos os *data marts* de uma organização [KIM98]. Especificamente, o *DW* pode ser considerado uma fonte de consulta de dados, que visa disponibilizá-los de maneira compreensível e otimizada, a partir de campos pré-calculados e de um modelo dimensional de dados. O *DW* serve como uma implementação física do modelo de dados de suporte à decisão organizacional, armazenando informações cruciais à tomada de decisão estratégica.

Já um ambiente de *data warehousing* é caracterizado por contemplar o processo de ETC e a manutenção dos dados do *DW*. O estabelecimento desse ambiente é útil pelo ponto de vista da integração de fontes de dados heterogêneas. Diversas organizações tipicamente armazenam e mantêm variados tipos de dados provenientes de grandes bases de informações heterogêneas, autônomas e distribuídas. Assim, a consolidação desses dados em um ambiente

único, de maneira estruturada e concisa, permite a aplicação de técnicas de consultas com o intuito de auxiliar a tomada de decisão, melhorar o entendimento do negócio e permitir a manutenção do histórico da organização.

2.4.1 Modelo Dimensional

Tipicamente, os dados em um *DW* estão estruturados segundo um modelo dimensional. Seus dados estão empacotados em um formato projetado para possibilitar mudanças e aumentar o entendimento e o desempenho das consultas. O modelo dimensional apresenta, como componentes básicos, tabelas dimensão e fato.

As tabelas dimensão representam as diferentes perspectivas de análise de um determinado fato. A maioria das dimensões contém atributos textuais, essenciais para a construção das consultas e agrupamento dos fatos. Logo, entende-se por dimensões as perspectivas de uma base de dados que possam gerar registros. Tais dimensões são basicamente organizadas em torno de um termo central, onde os registros são inseridos, representado pela tabela fato [KIM98].

As tabelas fato são consideradas as estruturas básicas e primordiais do modelo dimensional. As mesmas devem conter mensurações do negócio definidas como métricas. Na maioria das vezes os fatos apresentam métricas com valores aditivos e numéricos. Além disso, cada tabela fato apresenta relacionamentos do tipo muitos para muitos e seus atributos não precisam ser necessariamente métricas de *software*; pois podem representar qualquer atributo do negócio.

O que determina a escolha do modelo analítico dimensional a ser desenhado para o *DW* são as necessidades do negócio e as características de suas bases de dados. No que tange a essa definição, é possível representar diferentes tipos de modelos analíticos:

- **Modelo Estrela:** caracteriza-se essencialmente por conter uma única tabela fato e várias dimensões, cujas chaves compõem essa tabela.
- **Modelo floco de neve:** difere-se do modelo estrela por admitir que as tabelas dimensões possuam outros relacionamentos.
- **Modelo Constelação de Fatos:** de uma forma mais abrangente, permite dispor mais de uma tabela fato, as quais possuem suas dimensões, e tais dimensões, também podem admitir outros relacionamentos [HAN01].

A idéia fundamental do modelo analítico dimensional está relacionada com a possibilidade de representar dados de negócio na forma de cubos, onde as células contêm os valores das métricas e as arestas significam as dimensões dos dados. Tipicamente, em modelos dimensionais de negócio reais, é comum encontrar cubos com dimensionalidade maior que três, podendo variar seu tamanho entre quatro e quinze dimensões [KIM98].

2.4.2 Recursos *OLAP*

Os recursos *OLAP* (*On-Line Analytical Process*) permitem a realização de consultas em base de dados dimensionais e a apresentação dos seus resultados, textuais ou numéricos, de maneira clara e simplificada, considerando as diferentes perspectivas de análise e níveis de sumarização. Esses recursos admitem que os usuários do *DW* possam explorar os dados através de tabelas pivotantes, possibilitando uma apresentação tabular, e a interação por operações *OLAP* de *drill-down e drill-up*. Essas operações, quando implementadas de maneira correta, mesclam atributos hierárquicos ou não de todas as dimensões disponíveis.

Um aspecto relacionado com as operações *OLAP* que merece destaque é a possibilidade de executar consultas à base de dados sem a necessidade de editar código e sem ter conhecimento técnico aprofundado do negócio ou da interface *OLAP* utilizada. Por essas razões, os recursos *OLAP* apresentam as características ideais para oferecer informações diferenciadas, conforme distintos papéis organizacionais e níveis de gestão.

2.4.3 Processo de ETC

Para que os dados possam ser devidamente disponibilizados no ambiente de *data warehousing*, é necessário estabelecer um processo de ETC. Este tem por objetivo principal proporcionar, aos usuários do *DW*, dados estruturados, consistentes e de qualidade para oferecer suporte à tarefa de tomada de decisão de negócio. No contexto de PDS, a ETC de métricas de *software* é responsável por capturar, transformar e carregar os dados dos processos, finalizados e em execução, e dos artefatos produzidos e utilizados por eles. As seções subseqüentes abordam cada uma das etapas desse processo, detalhando as suas particularidades e os seus aspectos relevantes.

2.4.3.1 Extração

A extração representa o primeiro passo para inserir dados no ambiente de *data warehousing*. Extrair significa ler e entender as fontes de dados originais, copiando apenas as partes necessárias para uma área de armazenamento temporária. A partir desta última, os dados podem ser transformados conforme o modelo dimensional do *DW* alvo. As fontes originais podem apresentar localizações físicas distintas, estruturas de dados diferentes e ambientes de execução variados, caracterizando um ambiente heterogêneo. Além disso, essas fontes também podem conter um volume de dados bem maior do que se deseja extrair. Por essa razão, nem todas as informações mantidas nas bases de dados das ferramentas originais agregam valor ao negócio e precisam ser consolidadas e analisadas.

2.4.3.2 Transformação

Após a extração, os dados devem ser temporariamente mantidos em uma área de armazenamento especial, denominada *DSA* (*Data Staging Area*). No entanto, para que os mesmos possam ser consolidados e armazenados no *DW*, segundo o modelo dimensional, uma série de transformações precisa tipicamente ser realizada. A mesma inclui atividades para: (i) limpar ruídos; (ii) resolver domínios; (iii) solucionar domínios conflitantes, dados faltantes e formatar valores conforme o padrão pré-estabelecido; (iv) eliminar campos selecionados dos sistemas legados que não são úteis ao *DW*; (v) combinar fontes de dados, verificando a integridade entre as chaves primárias ou realizando combinação entre atributos não chave, incluindo a procura por textos equivalentes em códigos fonte de sistemas legados; e (vi) criar *surrogates* para cada registro das dimensões, com a finalidade de evitar a dependência com as chaves legadas e, dessa forma, reforçar a integridade referencial entre as tabelas fato e dimensão. A etapa de transformação pode ser considerada primordial, pois permite que dados brutos, de fontes distintas ou não, possam ser alterados e consolidados resultando em informações relevantes, concisas e de qualidade para o negócio. Assim, os valores obtidos tornam-se mais próximos dos reais e auxiliam à tomada de decisão.

2.4.3.3 *Data Staging Area* (*DSA*)

Segundo [KIM98], a preparação dos dados, efetuada em uma área de armazenamento temporária denominada *DSA*, é considerada uma das etapas mais importantes e críticas de um projeto de *DW*. Nessa área é executado um conjunto de processos para limpar, transformar, combinar, duplicar, arquivar e preparar os dados para serem utilizados pelo *DW*. Esses

processos são determinados por atividades simples de ordenação e processamento seqüencial, e em alguns casos, não precisam ser baseadas no modelo relacional. Para tanto, [KIM98] propõe um plano de dez passos para a criação de uma *DSA*:

1. Criar um plano de alto nível, por intermédio de um esquema, ilustrando o fluxo de dados da fonte para o destino, sendo o mesmo representado em uma única página.

2. Testar, escolher e implementar uma ferramenta de *data staging*.

3. Diminuir a granularidade da tabela alvo, esboçando graficamente qualquer transformação e reestruturação de dados complexa, e ilustrando, também, o processo de geração de chaves *surrogates*.

4. Construir e testar a carga de uma dimensão estática. O objetivo principal deste passo é desenvolver uma infra-estrutura que suporte conectividade, transferência de arquivos e resolva problemas de segurança.

5. Construir e testar minuciosamente processos de mudanças para uma dimensão.

6. Construir e testar cargas nas dimensões restantes.

7. Construir e testar cargas nas tabelas fato históricas, incluindo substituição de chaves *surrogates*.

8. Construir e testar o processo de carga incremental.

9. Construir e testar a carga em tabelas agregadas e/ou carga em *MOLAP* (*Multidimensional Online Analytical Processing*).

10. Projetar, construir e testar a automação da aplicação de *staging*.

2.4.3.4 Carga

No final do processo de transformação os dados devem apresentar a forma adequada para que sejam inseridos no *DW*, conforme o modelo dimensional. Assim, os mesmos podem ser devidamente carregados da *DSA* para o *DW*. Com relação à carga, a maioria das implementações de processos de ETC, em ferramentas líderes de mercado, realiza a mesma de maneira incremental, mesmo que os dados envolvidos sejam *snapshots* extraídos somente

uma vez por mês. Nesse sentido, é muito mais eficiente carregar incrementalmente apenas os registros que apresentaram modificações ou que foram adicionados após a última carga. Usualmente, a carga incremental é baseada em uma data de transição ou algum outro tipo de indicador. A data de última carga também é considerada importante para realizar a manutenção do processo de carga [KIM98].

DW é tipicamente construído para conter dados que representam fatos encerrados. Por essa razão, não precisam se preocupar com acontecimentos em andamento, caracterizados por apresentar alterações constantes nos valores armazenados. No entanto, para manter um histórico adequado dos fatos em andamento, é necessário prover informações pertinentes no *DW*. Por essa razão, o processo de carga deve estabelecer regras para lidar com atributos que sofrem alguma alteração em relação aos valores anteriormente armazenados no *DW*.

2.5 Arquitetura Orientada a Serviços (AOS) + *Web Services (WS)*

A AOS pode ser definida como uma coleção de serviços, que apresentam baixo acoplamento entre si, sendo que os mesmos encapsulam componentes de *software* reutilizáveis e se comunicam com outros serviços. Essas interações são realizadas através de tecnologias de comunicação, que podem envolver uma troca de dados entre dois ou mais serviços que estejam coordenando alguma atividade [KEE04].

WS são considerados a tendência de tecnologia para desenvolver serviços que implementam as funções de negócio, segundo uma AOS. Isso está relacionado com a padronização do protocolo e da linguagem de comunicação utilizada por esses serviços. Estudos, como [CAS02a] [LEY03] [AAL03], declaram que a utilização de *XML* como a linguagem padrão de comunicação entre aplicações, permite que a troca de informação por intermédio de mensagens *SOAP* (*Simple Object Access Protocol*) possa ocorrer em ambientes heterogêneos e de maneira simples. A ubiqüidade do protocolo de comunicação e de transporte, utilizado por essas mensagens, também contribui significativamente para disseminar sua utilização. No entanto, cabe salientar que os *WS* não foram desenvolvidos para apresentar uma visão lógica orientada a negócios [WON06], sendo essa carência suprida pelo uso de uma AOS.

A utilização de *WS*, definida conforme uma AOS, permite que as aplicações encapsuladas pelos serviços, e eventualmente escritas em linguagens distintas, possam ser executadas em servidores diferentes, sem a necessidade de considerar como cada um desses serviços foi desenvolvido. Por intermédio da tecnologia de *WS* é possível mapear os processos

de negócio em serviços e, assim, publicá-los e descobri-los dinamicamente, tanto dentro dos limites de uma corporação como fora deles. Para isso, os *WS* podem serguir o modelo teórico definido pela AOS, o qual é baseado em três entidades e três operações básicas, sendo a combinação entre essas entidades a responsável por esse paradigma de comunicação entre aplicações.

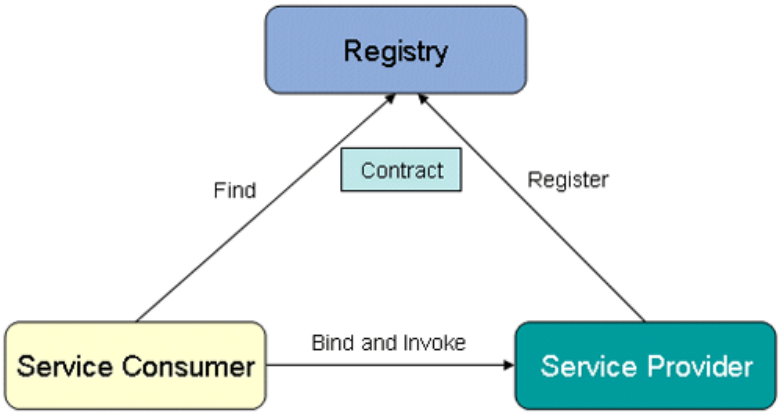


Figura 2 – Principais componentes de uma AOS.

A Figura 2 ilustra as principais entidades de uma AOS (*Service Provider*, *Registry* e *Service Consumer*) e as operações básicas entre elas (*Register*, *Find* e *Bind and Invoke*):

- ***Service Provider* (Provedor de Serviço):** entidade que cria os *WS*, descrevendo os serviços em um formato padrão, compreensível a qualquer cliente que queira utilizá-los. Além disso, publica através da operação *Register*, os detalhes desses serviços em um registro público (*Registry*), disponível a todos os interessados.

- ***Registry* (Registro de Serviço):** local onde os provedores registram seus serviços, sendo considerado uma central de serviços, onde os clientes podem pesquisá-los e encontrá-los conforme suas necessidades.

- ***Service Consumer* (Cliente do Serviço*):*** é a entidade que solicita o serviço, por intermédio da operação *Find*, ao Registro de Serviço que, por sua vez, encontra um provedor de serviços disponível e repassa o pedido para esse último, utilizando a operação *Bind and Invoke*.

Os *WS*, baseados nessas entidades que compõem a AOS, são publicados para os clientes na *web* quando são registrados no diretório *UDDI* (*Universal Description, Discover and Integration*). A estrutura e o modo de funcionamento dos componentes *XML*, *SOAP*,

WSDL (*Web Services Description Language*) e *UDDI*, responsáveis pelos *WS*, são descritos nas próximas seções.

2.5.1 *eXtensible Markup Language* (*XML*)

A utilização de *XML* [W3C06a] como linguagem padrão para a comunicação dos *WS* foi de extrema importância para a sua disseminação e aceitação. A adoção desse padrão garante simplicidade e heterogeneidade de linguagens de programação e plataformas, devido ao fato de ser muito próximo à linguagem natural, extensível e não requerer conhecimentos aprofundados para sua utilização. Além disso, garante flexibilidade à troca de mensagens entre os serviços [CAS02a], pois é considerada uma linguagem descritiva baseada em etiquetas (*tags*), que podem ser interpretadas conforme a necessidade do cliente, permitindo que a mesma informação possa ter significado distinto para duas ou mais aplicações. Cabe salientar que, para isso, elas devem respeitar um mesmo esquema (*XML-schema*) conhecido pelas partes envolvidas, o qual é determinado durante o contrato do serviço. Esse contrato é definido como a ligação entre o cliente e o *WS*, e sua validade é expirada quando todas as iterações, realizadas pelas trocas de mensagens, são executadas. Já os *XML-schemas* [W3C06d] exprimem vocabulários e definem as regras para escrever os documentos *XML*, bem como sua estrutura, seu conteúdo e a sua semântica.

2.5.2 *Simple Object Access Protocol* (*SOAP*)

A comunicação entre os *WS* é feita através do protocolo *SOAP*, que tem por objetivo organizar as informações escritas em *XML* de maneira estruturada, permitindo a troca de mensagens entre as aplicações [ALO04]. No entanto, *SOAP* apenas define o formato dessas mensagens, deixando a semântica por conta das aplicações.

O protocolo *SOAP* [W3C06c] baseia-se na linguagem *XML*, diferentemente do formato binário utilizado por outros protocolos de comunicação de sistemas distribuídos, como: *RMI*, *DCOM* e *CORBA*, os quais usam *RPC* (*Remote Procedure Call*) para a comunicação entre seus objetos. As chamadas RPC não são facilmente adaptáveis ao padrão *web*, visto que o protocolo *HTTP* não foi desenvolvido para dar suporte às aplicações distribuídas. Porém, *SOAP* permite empacotar chamadas *RPC* dentro do conteúdo das mensagens trocadas entre as aplicações. Dessa forma, possibilita, além da transmissão de dados, chamadas e respostas de métodos de *RPC* entre o cliente e o servidor *SOAP*.

O conceito de *SOAP* é simples: se uma única conexão é utilizada para chamar uma função ou um objeto de um método, disponibilizado por um servidor, esse servidor não deve se preocupar com o tipo do cliente que realizou o chamado, desde que os dados que estão sendo transferidos pela conexão atendam os padrões de formatação definidos pelo protocolo *SOAP*.

SOAP foi desenvolvido para utilizar o padrão *HTTP*, considerado o padrão de comunicação mais popular para a utilização de *WS*. Logo, apresenta tráfego livre entre *firewalls* e servidores *proxy*, sendo suportado por todos os servidores e navegadores *web*. Por esse motivo, *SOAP* caracteriza-se como um protocolo confortável para a comunicação entre aplicações remotas. Além do protocolo *HTTP*, as mensagens *SOAP* também podem ser transmitidas por *FTP* e *SMTP*. O cabeçalho das mensagens *SOAP* especifica detalhes para o protocolo de transporte, enquanto que o corpo, tipicamente, contém os parâmetros utilizados para chamar os métodos dos serviços e os seus resultados.

Segundo [SID02], um servidor *SOAP* em Java pode ser considerado um *listener* (um ouvinte permanente), executado dentro de um servidor *web* onde estão hospedados os *WS*, que ficam aguardando chamadas de clientes. A Figura 3 ilustra esse servidor (*Web Server*), que apresenta três componentes principais: um Gerenciador de Serviços, um Tradutor *XML* e uma Lista de Serviços. O Gerenciador de serviços recebe a chamada do cliente (*SOAP Request*) e verifica se o serviço requisitado por ela encontra-se na lista de serviços disponíveis no servidor web. Caso a busca tenha sucesso, o Gerenciador de Serviços encaminha a chamada ao Tradutor *XML*, que é o responsável por processar a mensagem *SOAP*, para que a chamada do cliente possa ser compreendida pela aplicação que implementa as funcionalidades do *WS*. Finalizada a execução da aplicação, o seu resultado é repassado ao tradutor para que seja incorporado na mensagem de resposta (*SOAP Response*), a qual é encaminhada ao Gerenciador do Serviço, que a transmite via *HTTP* ao cliente.

2.5.3 *Web Services Description Language* (*WSDL*)

A especificação *WSDL*, definida em *XML*, permite que os provedores de *WS* possam disponibilizar suas interfaces de acesso de forma padronizada [NEW02]. O conteúdo dessa especificação consiste em todas as informações necessárias para que o cliente possa acessar os *WS*, tais como: localização, métodos disponíveis e tipos de dados envolvidos. Uma *WSDL* pode ser considerada um contrato entre o cliente e o provedor do serviço, definindo o que pode ser feito, onde o serviço está hospedado e como pode ser acessado. Dessa forma, o

cliente não precisa ter conhecimento prévio da localização dos serviços, da maneira como foram implementados, da linguagem e da plataforma utilizada [ALO04]. Portanto, a *WSDL* garante a interoperabilidade entre linguagens e plataformas, e sua padronização é definida pela W3C *Web Services Description Working Group* [W3C06c], composta por organizações como Sun, *IBM*, *HP*, entre outras.

Figura 3 - Estrutura interna do servidor *SOAP* em Java [SID02].

As informações que compõem a *WSDL* estão dispostas na forma de metadados padronizados, os quais são organizados na forma de etiquetas (*tags)*. Essa estrutura permite que um *parser XML* possa percorrer o arquivo *WSDL* e reconhecer as informações que devem ser extraídas para a especificação do conjunto de mensagens *XML SOAP* do cliente. Os valores dessas informações são armazenados em variáveis locais, que podem ser utilizadas para gerar outras aplicações. Concluída essa etapa de extração e geração da aplicação o cliente está apto para se conectar ao *WS*.

O documento *WSDL* é constituído por duas partes denominadas: abstrata e concreta, também chamadas de não-funcionais e funcionais, respectivamente (Figura 4). A descrição concreta é composta por elementos que são orientados para vincular fisicamente o cliente ao serviço, enquanto que a abstrata contém elementos orientados para descrever as competências do *WS*.

A parte abstrata tem a responsabilidade de definir o que um *WS* pode fornecer em termos de funcionalidade e é formada pelos seguintes elementos:

- *Types* apresentam a estrutura dos tipos de dados que compõem o serviço;

- *Messages* apresentam a comunicação efetiva entre o cliente e o serviço, definindo quais os serviços que devem ser executados;

- *Operations* são chamadas de serviços, onde as operações possíveis são: requisição/resposta (cliente faz uma requisição e o *WS* responde), solicitação/resposta (*WS* envia uma mensagem ao cliente), sentido único (o cliente envia uma mensagem ao *WS*, mas não espera resposta) e notificação (*WS* envia uma mensagem ao cliente, sem esperar uma resposta). Interações síncronas são definidas através de operações requisição/resposta ou solicitação/reposta, enquanto que interações assíncronas são estabelecidas utilizando sentido único ou notificação;

- *Port Types* são utilizados para informar ao cliente todos os serviços disponíveis no *WS*.

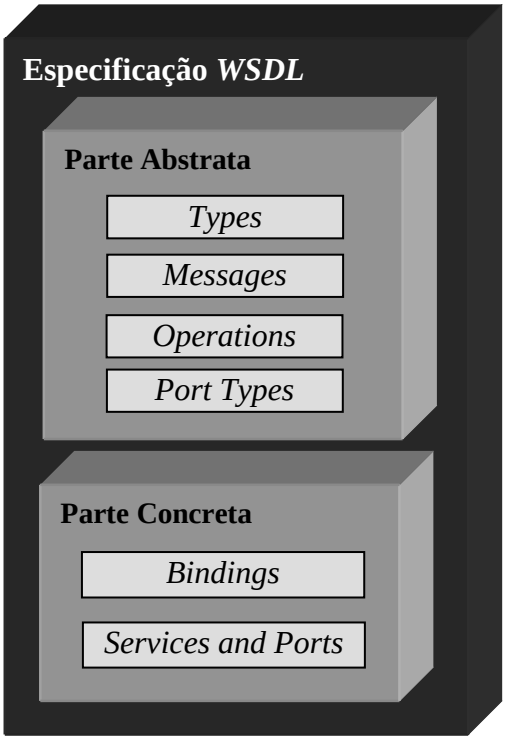


Figura 4 - Especificação do documento *WSDL* [ALO04].

O restante do documento *WSDL* é composto pela parte concreta, cuja finalidade é estabelecer ligações físicas entre os *WS*, subdividida em *bindings* e *services and ports*.

- *Bindings* realizam o vínculo entre os atributos concretos e os abstratos no documento *WSDL*, fornecendo informações, tais como: o protocolo e o endereço do *WS*.

- *Services* e *Ports* apresentam a porta real e o *IP* do *WS* que estão sendo representados pelo documento *WSDL*.

2.5.4 *Universal Description, Discover and Integration* (*UDDI*)

Os *WS* podem ser disponibilizados por meio de registros de Internet, estruturados conforme alguns padrões, chamados *UDDI*s [ALO04]. A estrutura desses padrões possibilita que o cliente de serviços possa ir ao diretório e acessar o serviço que melhor satisfaz a sua necessidade em tempo de execução. As regras e padrões que definem o *UDDI* são estabelecidos pela *Organization for Advancement Structured Information Standards* (*OASIS*) [OAS06].

O *UDDI* é um meio de publicar e encontrar serviços. Sua utilização tem um papel fundamental na disponibilização dos *WS*, pois proporciona, de forma centralizada, a descoberta de serviços disponíveis, suas localizações e a forma de interação com os mesmos. A utilização desse centralizador permite que novas versões de um determinado serviço possam ser disponibilizadas, sem que o sistema distribuído seja interrompido. Além disso, pode-se permanecer com os serviços de versões anteriores que satisfaçam determinados clientes e com serviços novos que atendam especificações recentes de diferentes clientes.

A possibilidade de reutilização de código também representa uma parcela importante no que se refere à agilidade frente às modificações exigidas pelo negócio. Como os serviços são considerados aplicações bem definidas e com baixo acoplamento, podem ser compostos ou alterados para dar suporte às novas necessidades, de forma mais otimizada, por intermédio da simples publicação e edição de novos serviços.

2.5.5 Interação entre os padrões que compõem os *WS*, segundo uma AOS

O diagrama, ilustrado pela Figura 2, apresenta a interação entre os componentes que constituem uma AOS, especificados nas seções anteriores deste capítulo, e os padrões utilizados pelos *WS*. O ciclo básico de funcionamento dos *WS* (Figura 5) inicia com sua disponibilização em um provedor de serviços (*Service Provider*), na forma de um servidor *web*. O primeiro passo (1) para oferecer esses serviços aos clientes ocorre com sua publicação (*Publish*) em um repositório centralizado (*Service Repository*), implementado por meio de um repositório *UDDI*. Após a publicação, todas as informações necessárias para encontrar e estabelecer contratos entre serviços e clientes estão disponíveis. Independentemente da publicação dos serviços, os clientes podem pesquisar (*Find*) (2) por eles através de buscas no repositório *UDDI* que, por analogia, pode ser comparado às páginas amarelas de um guia telefônico, onde se encontram as informações dos serviços disponíveis, dos seus fornecedores e da sua localização. Quando o cliente encontra o serviço que atende às suas necessidades,

recebe os dados necessários para estabelecer um contrato com o fornecedor do serviço (3). O cliente, de posse desses dados, estabelece o contato com o *WS* que apresenta as funcionalidades desejadas (*Bind*) (4) através de mensagens *SOAP*, que contêm a localização do serviço, bem como seus métodos, seus tipos de dados e os seus parâmetros. O processador *SOAP* (*listener*), presente no servidor web, processa essa mensagem enviada pelo cliente, através dos dados presentes no seu cabeçalho e procura o serviço na lista de serviços. Quando o encontra, estabelece a ligação e repassa os dados da sua chamada. Assim, a aplicação que constitui o serviço é executada e os resultados são repassados ao processador *SOAP*, que é responsável por compor a mensagem de resposta e enviá-la ao cliente (5). Essas interações entre os clientes e os serviços são feitas diretamente, conforme a quantidade de iterações necessárias para que o serviço consiga concluir o objetivo. Esses serviços também podem ser chamados por outros serviços, não sendo essa ação exclusiva dos clientes.

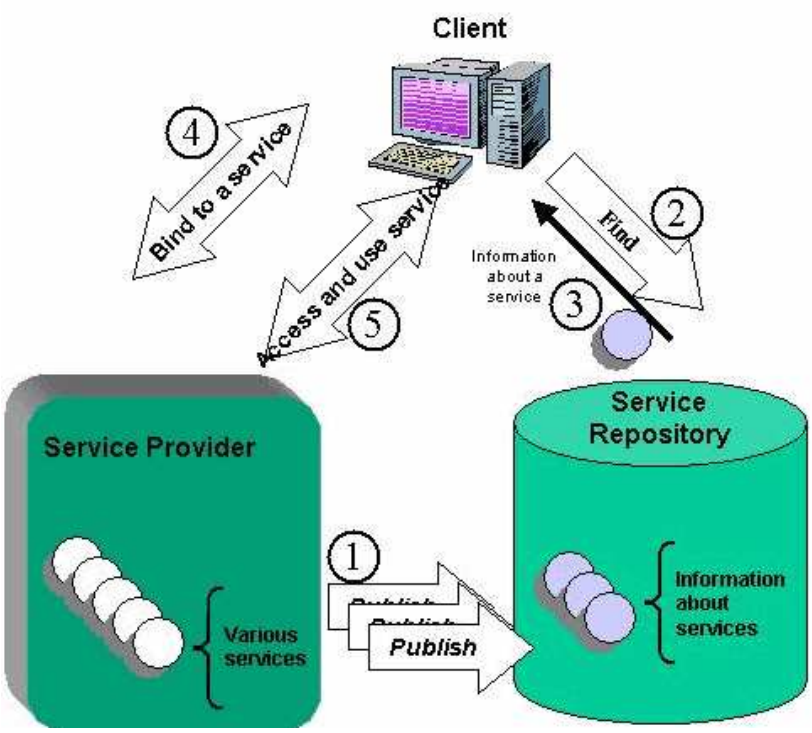


Figura 5 - Visão geral dos *WS* [ORE06Erro! Fonte de referência não encontrada.**].**

A maneira como os *WS* são disponibilizados permite que os contratos entre clientes e fornecedores de serviços possam ser feitos dinamicamente, conforme as necessidades dos clientes e as facilidades oferecidas pelos fornecedores de serviços. Além disso, alterações e novas versões podem ser disponibilizadas em paralelo com os sistemas existentes, proporcionando flexibilidade e adaptabilidade do negócio. Outro ponto que merece destaque é a padronização de comunicação. Com ela, os clientes não precisam ter conhecimento da

linguagem e do código envolvido na aplicação que oferece o serviço, fator que garante a heterogeneidade de ambiente e a interoperabilidade de aplicações.

3 DESCRIÇÃO DO CENÁRIO

Este capítulo descreve o cenário onde esta pesquisa está inserida, de maneira sistemática, por intermédio de três dimensões ortogonais: 1) nível de mensuração, 2) atividades de supervisão, e 3) plataforma computacional. Para tanto, são relatados os aspectos específicos de cada uma dessas dimensões e, posteriormente, é realizada uma leitura das suas possíveis interações, buscando caracterizar àqueles aspectos essenciais para coletar métricas, e oferecer suporte à mensuração da qualidade de *software*. Faz-se uso dos termos e descrições referentes a métricas de *software*, feitas no Capítulo 2.

3.1 Dimensões da Mensuração da Qualidade de *Software*

No desenvolvimento de *software*, a qualidade do produto está diretamente relacionada à qualidade do seu processo de desenvolvimento [KHO01]. A norma *ISO* 9000 [ABN05] define qualidade como o grau em que um conjunto de características inerentes a um produto, processo ou sistema cumpre os seus requisitos inicialmente estipulados. Nesse sentido, foram criados modelos de qualidade para *software*: *CMMI* [SEI06], *SPICE* (*Software Process Improvement and Capability Determination*) [EMA97], MPS.BR (Melhoria de Processos do *Software* Brasileiro) [SOF07]. Os mesmos apresentam processos bem definidos, práticas e objetivos, gerais e específicos, organizados em níveis de maturidade e competência. Na tentativa de minimizar riscos e adquirir produtos de qualidade, com preços justos e dentro dos prazos previstos, muitos clientes buscam empresas de desenvolvimento de *software* certificadas, segundo esses modelos. Por sua vez, essas empresas, para atender as exigências dos clientes e continuar competitivas no mercado, também acabam optando, freqüentemente, por obter certificações de qualidade.

Tanto esses modelos de qualidade ([SEI06] [EMA97] [SOF07]), como a norma 1061 [IEE98] e Stephen Kan [KAN03], afirmam que a mensuração da qualidade do *software* pode ser efetuada por intermédio de métricas (Seção 2.2.1), definidas segundo um programa de métricas (Seção 2.2.2.1). Para sistematizar a discussão, sobre a mensuração da qualidade de *software*, optou-se por esquematizar os aspectos envolvidos na sua realização em três dimensões ortogonais: 1) nível de mensuração, 2) atividades de supervisão, e 3) plataforma computacional. A primeira dimensão apresenta os níveis de mensuração propostos pelos modelos de qualidade [SEI06] [SOF07]. A segunda contempla as atividades básicas de gestão adotadas por diferentes segmentos de desenvolvimento de projetos (*e.g.* construção civil,

construção naval) e sugeridas por institutos reconhecidos, como: *Software Engineering Institute* (*SEI*) [SEI06] e *Project Management Institute* (*PMI*) [PMI04]. A terceira trata de aspectos das empresas de desenvolvimento de *software* que influenciam na realização da solução computacional de apoio à mensuração da qualidade. A esquematização dessa última dimensão é baseada em estudos anteriores [BEC06] [CUN05] e no conhecimento de um ambiente de desenvolvimento de *software* de grande porte.

3.1.1 Nível de mensuração

A mensuração da qualidade de *software* pode ser realizada em diferentes níveis: (i) equipe (*e.g.* qualidade, teste, desenvolvimento); (ii) projeto; (iii) organizacional; (iv) interorganizacional (*e.g.* operações de *software* globalizadas). Porém, os modelos de qualidade de *software* ([SEI06] [EMA97] [SOF07]) tratam apenas os níveis de projeto e organizacional, simplificação convenientemente adotada nesta pesquisa.

A realização da mensuração da qualidade de *software* depende da coleta de métricas do seu ambiente de desenvolvimento, tendo auxílio computacional ou não para a sua gestão. Assim, as métricas diretas podem estar originalmente armazenadas na forma: (i) *adhoc*, em planilhas ou documentos texto; (ii) integrada, a partir de ferramentas de automação de escritório (*e.g.* pacote *Office* da *Microsoft*); ou (iii) automatizada, através de ferramentas, dedicadas ou personalizadas, orientadas à gestão, que apresentam tratamento especial para as métricas (*e.g. MS Project*, *Project and Portfolio Management*, *IBM Rational ClearQuest*). No primeiro caso, as métricas encontram-se apenas documentadas em arquivos, sem nenhuma estrutura ou organização pré-definidas. Já no segundo, as mesmas também estão documentadas em arquivos de forma estruturada e organizada, tipicamente seguindo uma padronização estabelecida (*document template*). Por fim, no terceiro caso, as métricas estão localizadas em bases de dados das respectivas ferramentas.

Os modelos de qualidade de *software*, *CMMI*3 e MPS.BR Nível E, exigem que além da coleta das métricas, a partir das diferentes formas de armazenamento apresentadas, seja realizada a sua consolidação em um repositório central único, tanto no nível do projeto como no da organização. Com base nesse último é possível obter uma visão comparável e unificada dos diferentes projetos, objetivando auxiliar os diversos níveis de gestão (*e.g.* organizacional, projeto) e papéis organizacionais (*e.g.* gerente de projeto, gerente de *software*, gerente de qualidade). A existência de um repositório central permite a manutenção da história dos dois níveis. Dessa forma, as métricas de *software* podem ser utilizadas como fonte de dados para a

realização de estimativas e para o planejamento inicial de novos projetos. Além disso, o repositório também permite compartilhar boas práticas de desenvolvimento de *software* entre os projetos da organização e apresentar aos clientes métricas sobre o desempenho do seu projeto ou de experiências anteriores.

Em alguns casos, os repositórios de métricas baseiam-se em ambientes de *data warehousing* (Seção 2.4) ([BEC06] [GOL04] [GRI04] [PAL03]), estruturados segundo um modelo dimensional (2.4.1). Vale ressaltar que a definição desse ambiente deve incluir um processo de ETC (Seção 2.4.3), o qual faz parte da etapa de preparação dos dados, considerada crítica por [KIM98]. Também é importante salientar que a forma de armazenamento das métricas está intimamente relacionada com a definição de processo.

3.1.2 Atividades de Supervisão

As atividades de análise consistem no estudo sobre o histórico de métricas através de técnicas estatísticas, mineração de dados, geração de gráficos e verificação de indicadores de desempenho, tendo em vista a utilização dos seus resultados para um melhor entendimento do próprio projeto, formular planos de melhorias e ações corretivas no mesmo, ou em projetos que apresentem características semelhantes. Por exemplo, pode-se analisar o percentual de retrabalho para corrigir defeitos e buscar descobrir a causa raiz dos mesmos e, com isso, criar um plano de melhoria para reduzir os principais fatores responsáveis pela geração de defeitos.

Já a monitoração tem por objetivo oferecer informações do andamento dos projetos em relação ao que foi inicialmente planejado, a partir de um acompanhamento regular do planejamento e buscando detectar desvios significativos [SEI06]. Esse acompanhamento pode ser efetuado por intermédio de técnicas específicas de monitoração (*e.g. EVA*, Seção 2.3) e deve ser realizado de modo a oferecer informações, úteis e atualizadas, que auxiliem na tomada de decisão, para que ações corretivas possam ser executadas, no momento certo. Por exemplo, quando uma determinada tarefa encontra-se em atraso e a sua finalização é pré-requisito para o início de outras tarefas, o gestor pode optar por alocar mais recursos nesta tarefa atrasada, ou transferi-la para outro recurso mais qualificado e/ou disponível, para que a mesma não cause impacto nas demais e não ocasione um atraso do prazo de entrega e aumento do orçamento, ou, pelo menos minimizar esse impacto.

Por sua vez, a previsão também está relacionada com o estudo de dados passados. Porém, ela objetiva a realização de estimativas confiáveis e próximas de valores reais, bem como na detecção de desalinhos de objetivos, além da descoberta da causa raiz de falhas dos

produtos. Os resultados da previsão podem ser oferecidos através de modelos, ditos preditivos, produzidos ou por técnicas probabilísticas ou por técnicas de mineração sobre as métricas de *software*. Dessa forma, os gestores podem se beneficiar do histórico de mensurações para melhorar suas estimativas, buscando atingir uma maior precisão dos prazos e custos. Por exemplo, com base no número de defeitos de versões anteriores, no esforço associado para a sua correção e no tamanho do seu código, técnicas preditivas podem ser aplicadas para estimar o número de defeitos da versão corrente, podendo dessa forma alocar horas de projeto em atividades de correção com uma maior precisão e, conseqüentemente, estimar custos e prazos mais próximos dos reais.

3.1.3 Plataforma Computacional:

A definição de uma arquitetura computacional para a automação, parcial ou total, da mensuração de qualidade de *software*, precisa tratar aspectos relacionados com: (i) a latência e a freqüência de coleta das métricas, (ii) o ambiente heterogêneo de desenvolvimento de *software*, e, principalmente, (iii) a intrusão do processo de captura das métricas. A seguir são discutidos cada um desses aspectos.

3.1.3.1 Latência e Freqüência de Coleta das Métricas

A latência na coleta de uma métrica consiste na diferença entre o instante de tempo em que ocorre o seu evento gerador e a disponibilização da mesma aos gestores. Dessa forma a latência pode ser vista como o tempo total consumido pelas seguintes ações: lançamento do evento ocorrido pelo recurso (integrante do projeto), execução da coleta das métricas, consolidação das mesmas no repositório e disponibilização dos seus resultados. Já a freqüência da coleta representa a periodicidade com que essa coleta das métricas é realizada no ambiente de desenvolvimento de *software*.

É importante ressaltar que a latência e a freqüência, definidas pela solução de automação, devem considerar algumas particularidades desse ambiente de desenvolvimento. Os valores de algumas métricas diretas, como esforço real e número de defeitos encontrados, tipicamente podem ser informados pelos recursos, ao término de um determinado evento. No caso do esforço, por exemplo, para cada tarefa planejada são atribuídos um ou mais recursos que devem lançar, em um registro de banco de horas (em uma base de dados ou não), o tempo consumido para a realização dessa tarefa. Esse lançamento pode ser realizado logo ao término da tarefa, no final do turno ou do dia de trabalho ou, até, ao término da semana, conforme a

política da organização. Conseqüentemente, a definição da freqüência de coleta deve considerar essas particularidades da latência para que os dados capturados representem uma situação correta das métricas, no instante em que foram coletadas. Nesse sentido, considerou-se que: (i) a latência pode ser nula, (ii) é medida em unidade de tempo e (iii) pode apresentar grandezas na ordem de segundos, minutos, horas, dias, semanas ou meses. Por sua vez, a freqüência é medida em número de vezes por unidade de tempo, onde esta última pode apresentar as mesmas ordens de grandeza da latência. Também vale ressaltar que o tempo de execução do processo de coleta das métricas é crucial para a determinação de ambas. A realização de tarefas laboriosas e manuais para coletar, consolidar e disponibilizar as métricas pode sobrecarregar e onerar o projeto, interferindo diretamente na definição da sua freqüência.

Com base no que foi exposto, conclui-se que a latência e a freqüência estão intimamente relacionadas. Contudo, para um melhor entendimento da contribuição desta pesquisa, optou-se por discutir as questões referentes à latência e à freqüência de maneira individual, adotando para a latência, as seguintes ordens de grandeza: horas, dias e meses. Horas, significando que ao término da tarefa ou, no máximo, ao final do turno/dia de trabalho, o recurso lança valores para as métricas no ambiente computacional e que essas passam a estar disponíveis. Dias, significando que ou o lançamento das métricas por parte dos recursos se dá com um espaçamento maior ou que a coleta é feita mais espaçadamente (uma vez por semana, por exemplo). Meses, significando que somente ao final de cada mês é que as métricas são consolidadas no repositório, mesmo que o lançamento das mesmas seja feito em intervalos menores. Com relação à freqüência foi adotada a mesma ordem de grandeza para a unidade de tempo.

3.1.3.2 Ambiente Heterogêneo de Desenvolvimento de *Software*

Com relação ao ambiente heterogêneo de desenvolvimento de *software*, o mesmo é conseqüente da diversidade de: (i) modelos de processo de *software*, (ii) ferramentas de apoio, (iii) tipos de projetos, e (iv) níveis de isolamento.

- ***Modelos de Processo de Software*:**

A Engenharia de *Software* propõe diversos modelos de processo, como: cascata, iterativo, espiral, unificado, entre outros (Seção 2.1.1), sendo que cada um deles pode apresentar ciclos de vida, fases e iterações distintas. Esses modelos servem de base para a realização do planejamento do projeto. Tipicamente é feita uma planificação dos mesmos em

função do tempo, representada por um cronograma, onde as linhas definem as tarefas que devem ser realizadas e as colunas as métricas diretas (*e.g.* esforço, custo, data inicial e final) ou demais informações necessárias para a condução do projeto.

Os modelos de processo interferem diretamente na ordem de execução (*e.g.* data inicial e final) e na disposição das tarefas no cronograma, originando uma estrutura hierárquica. No caso da organização estudada (Capítulo 5), diversas tarefas compõem uma fase, que por sua vez constituem uma iteração, que podem ser agrupadas em uma versão pertencente a um determinado projeto. Vale ressaltar que essa estrutura também pode ser modificada, conforme as necessidades dos projetos e as exigências dos clientes. Por essa razão, esses modelos e suas adaptações interferem diretamente na estrutura dimensional de armazenamento das métricas e no seu processo de coleta. Assim, quando essa heterogeneidade é constatada, é necessário considerar as diferentes estruturas de dados para extrair as métricas, bem como definir uma série de transformações para consolidá-las, segundo as diferentes perspectivas de análise permitidas pelos modelos de processo (*e.g.* fase, iteração, versão).

Logo, quanto maior a diversidade de modelos de processo e estruturas de cronograma, mais heterogêneo se torna o ambiente de desenvolvimento. Entretanto, essa diversidade não precisa necessariamente existir, e as questões mencionadas passam a não ser relevantes. Para o desenvolvimento desta pesquisa, assume-se que os modelos de processo podem apresentar o seguinte domínio: homogêneo ou heterogêneo, sendo que o segundo é caracterizado pela presença de no mínimo dois modelos de processo e de estruturas de cronograma distintos.

- ***Ferramentas de apoio:***

Atualmente, o mercado dispõe de uma diversidade de ferramentas para apoio à gestão, que permitem o armazenamento das métricas, como: i) *MS Project* [MIC07] e *Project and Portfolio Management* (*PPM*) [MER07] (cronograma); ii) *IBM Rational Clear Quest* [IBM07a], *Bugzilla* [THE07], *Mantis* [MAN07] e *Clarity* [KOR07] (acompanhamento e controle de defeitos); iii) *Borland CaliberRM* [BOR07] e *IBM Rational RequisitePro* [IBM07b] (requisitos) entre outras. Além dessas ferramentas, também podem ser utilizados editores de documentos texto e planilhas eletrônicas.

As organizações podem adotar um conjunto padronizado de ferramentas, onde uma mesma métrica é armazenada de forma idêntica em todos os projetos, e o seu ambiente de desenvolvimento é considerado homogêneo por este aspecto. Porém, no cenário mais típico, os projetos podem ter autonomia para escolher um conjunto de ferramentas que melhor atenda

as suas necessidades e as do seu cliente, ou ainda desenvolver as suas próprias ferramentas de apoio. Assim, a mesma métrica pode ser armazenada em diferentes bases de dados ou arquivos, conforme as suas respectivas estruturas. Além disso, uma mesma ferramenta pode apresentar variações na sua estrutura de dados segundo a sua versão e, ainda, oferecer, aos seus usuários, a opção de personalizar campos, aumentando ainda mais a heterogeneidade das formas de armazenamento. Por essas razões, as ferramentas e os arquivos utilizados pelos projetos interferem diretamente na definição da coleta das métricas, exigindo, no pior caso, um método específico de captura e transformação, para cada ferramenta, para que as métricas provenientes de diferentes fontes possam ser comparáveis. Dessa maneira, a heterogeneidade, conseqüente da diversidade de ferramentas, contribui significativamente para aumentar a complexidade de uma eventual automação da mensuração ou o esforço despendido no caso de coleta manual. Para o desenvolvimento desta pesquisa adota-se que o ambiente de desenvolvimento pode ser homogêneo, ou heterogêneo no que se refere às ferramentas de apoio à gestão. A existência de pelo menos duas ferramentas para armazenar a mesma métrica, já caracteriza o ambiente como heterogêneo.

- ***Tipos de Projetos:***

Os projetos podem ser classificados segundo seu porte (pequeno, médio e grande), em função do seu tempo de duração e do seu número de recursos. Nesse sentido, é possível encontrar casos reais de projetos, considerados de pequeno porte (*e.g.* dez recursos em tempo integral, em um ano de duração), onde o seu acompanhamento é realizado de maneira pessoal e direta, a partir de reuniões diárias entre os gestores e a equipe de desenvolvimento. Já em projetos de maior porte (*e.g.* 50 recursos e anos de duração) esse modo de acompanhamento se torna impraticável, devido ao número de recursos e dependência entre as suas tarefas. Por essa razão, nesse último caso é primordial a utilização de algum suporte computacional para conseguir acompanhar o andamento do projeto. Percebe-se, portanto, que as necessidades de informação dos gestores são distintas, e as métricas utilizadas também, bem como, o tipo de suporte computacional e, conseqüentemente, a forma de armazenamento das métricas no ambiente do projeto. Todos esses aspectos interferem diretamente no processo de coleta das métricas.

Nesta pesquisa, por motivos de conveniência foram adotados dois portes de projeto: pequeno e grande. O primeiro deve tipicamente apresentar equipes de no máximo vinte recursos, duração na ordem de meses, sendo suficiente pouco suporte computacional, e o

segundo pode ter acima de cinqüenta pessoas, anos de duração e ferramentas dedicadas de apoio à gestão são necessárias.

- *Níveis de Isolamento:*

Em muitos casos, clientes podem determinar, para seus projetos, políticas de segurança e privacidade, ferramentas de apoio à gestão, padrão de documentos de requisitos, classificação de defeitos, entre outros, que devem ser obedecidos pelas organizações de TI. Particularmente, as políticas de segurança e privacidade podem interferir diretamente no nível de acesso às métricas, por conseqüência das configurações de rede nos projetos (*e.g.* exigir que a rede de comunicação do ambiente de desenvolvimento de seu projeto seja fisicamente isolada tanto do mundo externo (Internet), como dos outros projetos da mesma organização (Intranet organizacional)). Dessa forma, esta pesquisa caracteriza a heterogeneidade do nível de isolamento como: isolado ou compartilhado. No primeiro os projetos apresentam redes internas, fechadas, onde somente os recursos do projeto têm permissão para realizar autenticação, impedindo a comunicação entre projetos. Já no segundo, os projetos compartilham a rede da organização e seus dados podem ser acessados por recursos externos. Por essas razões, essas políticas de segurança e privacidade precisam ser consideradas durante o processo de coleta, consolidação e disponibilização das métricas, sendo crucial que a solução de automação consiga tratar as dificuldades derivadas do nível isolado.

3.1.3.3 Intrusão do processo de captura das métricas

Intrusão consiste no grau de envolvimento dos recursos do projeto durante a execução do processo de coleta das métricas, que tipicamente deve apresentar os seguintes passos: (i) capturar as métricas da sua fonte de origem, (ii) editá-las e transformá-las convenientemente, e (iii) armazená-las em um repositório central. Para tanto, primeiramente, é preciso consultar as diversas fontes de métricas, considerando suas respectivas formas de armazenamento (*adhoc*, integradas, ou estruturadas). Após, deve ser efetuada, quando necessária, uma série de edições manuais ou não, (*e.g.* padronização da classificação da severidade dos defeitos, tratamento das chaves de identificação dos registros que contêm as métricas, entre outras) ou inserções em uma interface específica, para que as métricas possam ser consolidadas de maneira consistente e comparável. E, por fim, é necessário armazená-las em um repositório central. Nesse sentido, quanto mais manual e laborioso for esse processo, maior o envolvimento e, conseqüentemente, o nível de intrusão.

Por motivos de simplificação, nesta pesquisa, são adotados apenas três níveis de intrusão: isento, baixo e alto. No primeiro, considerado ideal, não existem atividades manuais, o processo é completamente automatizado. No segundo, o processo deve ter um bom nível de automação, demandando apenas que pequenas atividades, dos passos citados anteriormente, sejam manuais (*e.g.* homologação dos resultados ou ativação de uma rotina de extração). No terceiro, a maioria das tarefas é realizada manualmente exigindo um grande envolvimento dos recursos.

3.2 Interação entre as Dimensões

A partir de uma leitura das três dimensões ortogonais é possível examinar as interações entre elas e destacar os aspectos básicos que devem ser atendidos por uma solução computacional de automação da mensuração da qualidade de *software*. Essas interações estão sintetizadas na forma de uma tabela (Tabela 1), que pode ser vista como uma planificação do cubo formado pelas dimensões anteriormente discutidas, onde: (i) as colunas centrais representam as atividades de supervisão (análise, monitoração e previsão); (ii) a coluna lateral da esquerda refere-se ao nível de mensuração enquanto que, a da direita, aos aspectos envolvidos pela plataforma computacional; (iii) as linhas contemplam os aspectos computacionais e os níveis de mensuração; e (iv) as células contêm os domínios que devem ser considerados pela solução. Além disso, cada domínio pode ser classificado por uma das seguintes escalas: (1) aceitável (√), pouco aceitável (±) e inaceitável (×), e (2) existente (S) ou inexistente (Ñ). A segunda escala diz respeito somente aos aspectos de heterogeneidade, considerando os dois níveis de mensuração. Já a primeira envolve os restantes.

Com base nas interações entre as três dimensões e nas classificações dos domínios, apresentadas na Tabela 1, é possível destacar (sombreado em azul) os domínios dos aspectos mais críticos, para efetuar a mensuração da qualidade de *software* nos diferentes níveis, oferecendo suporte às atividades de análise, monitoração e previsão, considerando as heterogeneidades do ambiente de desenvolvimento, e realizar as seguintes leituras:

1º interação: Latência × (Análise, Monitoração e Previsão) × (De Projeto, Organizacional)

A partir dessa interação, primeira linha da Tabela 1, percebe-se que para realizar análise e previsão são aceitáveis todos os domínios definidos para latência. No entanto, o mesmo comportamento não pode ser aplicado à monitoração. Essa aceita, no máximo, latência na ordem de dias, pois requer valores atualizados para as métricas, que reflitam a situação atual do ambiente de desenvolvimento. Por essa razão, é possível constatar que a

baixa latência, associada à monitoração, é obrigatória para oferecer suporte a todas as atividades, nos dois níveis de mensuração.

Tabela 1 – Tabela de Interação entre as Dimensões.

	Atividades de Supervisão				
Nível de Mensuração	**Análise**	**Monitoração**	**Previsão**	**Plataforma Computacional**	
De Projeto Organizacional	Horas (√) Dias (√) Meses (±)	Horas (√) Dias (±) Meses (×)	Horas (√) Dias (√) Meses (±)	**Latência**	
De Projeto Organizacional	Horas (√) Dias (√) Meses (±)	Horas (√) Dias (±) Meses (×)	Horas (√) Dias (√) Meses (±)	**Freqüência**	
De Projeto	Homogêneo (S) Heterogêneo (Ñ)	Homogêneo (S) Heterogêneo (Ñ)	Homogêneo (S) Heterogêneo (Ñ)	**Modelos de Processo**	**Heterogeneidade**
	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	**Ferramentas**	
	Pequeno (Ñ) Grande (Ñ)	Pequeno (Ñ) Grande (Ñ)	Pequeno (Ñ) Grande (Ñ)	**Projetos**	
	Isolado (Ñ) Compartilhado (S)	Isolado (Ñ) Compartilhado (S)	Isolado (Ñ) Compartilhado (S)	**Nível de Isolamento**	
Organizacional	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	**Modelos de Processo**	
	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	**Ferramentas**	
	Pequeno (S) Grande (S)	Pequeno (S) Grande (S)	Pequeno (S) Grande (S)	**Projetos**	
	Isolado (S) Compartilhado (S)	Isolado (S) Compartilhado (S)	Isolado (S) Compartilhado (S)	**Nível de Isolamento**	
De Projeto Organizacional	Isento (√) Baixo (√) Médio (√) Alto (±)	Isento (√) Baixo (√) Médio (±) Alto (×)	Isento (√) Baixo (√) Médio (√) Alto (±)	**Intrusão**	

2º interação: Freqüência × (Análise, Monitoração e Previsão) × (De Projeto, Organizacional)

Através da análise dessa interação, segunda linha da Tabela 1, constata-se que as atividades de análise e previsão aceitam freqüências de coleta na ordem de dias. Por sua vez para monitoração esse domínio é pouco aceitável, sendo ideal apresentar uma freqüência de horas, para que seja possível efetuar um acompanhamento regular do andamento do planejamento inicial. Portanto, é possível concluir que, entre as atividades, a monitoração é determinante na definição da freqüência de execução do processo de coleta das métricas.

3º Interação: Heterogeneidade × (Análise, Monitoração e Previsão) × De Projeto

A partir dessa interação, terceira linha da Tabela 1, conclui-se que não existe, no nível de projeto, heterogeneidade de: modelos de processo, projetos e isolamento. Portanto, o

cenário mais crítico dessa interação é caracterizado pela existência de heterogeneidade de ferramentas, no ambiente, para a coleta das métricas, podendo ser mais simples mensurar a qualidade em um cenário totalmente homogêneo.

4º Interação: Heterogeneidade × (Análise, Monitoração e Previsão) × Organizacional

Com base nessa interação, quarta linha da Tabela 1, verifica-se no cenário mais crítico a existência de todas as heterogeneidades, de todos os tipos de projetos e do nível isolado. Estudos anteriores [BEC06] [CUN05] afirmam que a definição de um processo de coleta de métricas, que contemple todos esses aspectos e mantenha a liberdade, dos projetos e dos clientes, não é considerada uma tarefa trivial. A heterogeneidade contribui significativamente para aumentar a complexidade dos aspectos da automação computacional.

5º Interação: Intrusão × (Análise, Monitoração e Previsão) × (De Projeto, Organizacional)

Por intermédio dos valores dos domínios, localizados na última linha da Tabela 1, conclui-se que, para oferecer suporte à análise e à previsão, é aceitável um processo de coleta de métricas com um nível alto de intrusão, já que a freqüência de captura pode variar na ordem de meses. No entanto, para suportar monitoração, que apresenta uma freqüência no mínimo diária, torna-se inaceitável realizá-lo com um nível alto de intrusão, tanto no âmbito do projeto, como no da organização. A freqüência diária reforça a necessidade de que a mensuração seja realizada de modo pouco intrusivo, exigindo o mínimo de esforço para o time de desenvolvimento. Além disso, não é visto com bons olhos afastar recursos do projeto, das suas atribuições normais para efetuar a captura das métricas, pelo custo envolvido. Essa tarefa não pode ser realizada por qualquer recurso, pois o mesmo deve ter conhecimento tanto das particularidades do ambiente do projeto como da organização. Portanto, como as métricas são úteis não apenas ao projeto, não é justo que somente o mesmo seja onerado.

Logo, conclui-se que o pior cenário envolve a mensuração da qualidade do *software* no nível organizacional, com atividades de análise, monitoração e previsão, em um ambiente heterogêneo de desenvolvimento, de maneira pouco intrusiva, considerando a latência e a freqüência na ordem de horas. Desse modo, a definição de um processo de mensuração da qualidade de *software* pouco intrusivo é fundamental para a sua aceitação no ambiente do projeto.

3.3 Considerações Sobre a Descrição do Cenário

Este capítulo apresentou a mensuração da qualidade de *software* a partir de três dimensões ortogonais e as suas possíveis interações. Para tanto, foi utilizada uma tabela (Tabela 1) que pode ser vista como uma planificação dessas dimensões e que serve de base para: (i) discutir os trabalhos relacionados apresentados no Capítulo 4; (ii) levantar os aspectos envolvidos na mensuração da qualidade de *software*, que compõem a tabela, em um ambiente real de desenvolvimento, relatado no Capítulo 5; (iii) mencionar quais desses aspectos são tratados pela solução descrita no Capítulo 6; e (iv) verificar se a mesma os atende de maneira adequada, a partir dos testes documentados no Capítulo 7.

4 TRABALHOS RELACIONADOS

Este capítulo apresenta os trabalhos, encontrados na literatura, que são relacionados com o tema de pesquisa e que buscam soluções para mensurar a qualidade de *software*, a partir de métricas e de um repositório central. Entre eles encontram–se *MMR* [PAL03], *BPI*/*iBOM* [CAS02a] [CAS05] e *SPDW* [BEC06].

Ao final de cada uma das próximas seções é apresentada uma tabela contendo os aspectos da solução de automação, para mensuração da qualidade de *software*, a partir das três dimensões ortogonais definidas na Seção 3.1. Essa tabela tem por base a estrutura da Tabela 1, porém, as suas células estão preenchidas, somente, com os domínios atendidos pela solução relatada na Seção 3.2, onde (-) indica o não atendimento.

4.1 *MMR*

A ferramenta *MMR* foi desenvolvida para atender às necessidades de gerência de projeto da *Enterprise Performance Unit* da *Ericsson Research*, Canadá [PAL03]. A mesma propõe um repositório genérico, flexível e integrado, para coletar, armazenar, analisar e reportar dados, baseados nos requisitos de mensuração sugeridos pelo modelo *CMMI*. A *MMR* apresenta visões compartilhadas relacionadas à análise de tendências e acompanhamento do PDS, por intermédio de um repositório. Este último tem suas informações disponibilizadas por consultas multidimensionais sobre os dados, através de cubos *OLAP*. A flexibilidade desse repositório é oferecida através do desenvolvimento de metadados, que provêm definições de medidas e as relações entre elas. A *MMR* busca apoiar a maturidade de organizações nos diferentes níveis do *CMMI*, com o respectivo suporte ao seu programa de métricas. Nessa ferramenta a disponibilização de informações aos usuários finais é realizada por um portal *web*, na forma de relatórios pré-definidos e contendo alguns indicadores, sendo que novos relatórios podem ser desenvolvidos a partir de linguagem *SQL* ou através de recursos *OLAP*. Com relação à coleta das métricas, a *MMR* apresenta um método manual, onde os dados são inseridos individualmente, a partir de uma interface *web.*

A Figura 6 ilustra a arquitetura de *software* da *MMR* responsável pelo oferecimento das suas funcionalidades. Indicadores e Tendências têm por objetivo mensurar as questões de negócio definidas pela organização, respondendo como está o seu desempenho. Os mesmos podem ser divididos segundo quatro áreas distintas e providos pelo componente *Management*

Indicators & Trending. Dessa forma, pretende-se satisfazer as necessidades da gerência, apresentando informações baseadas em relatórios e gráficos pré-definidos disponibilizados na *web*, para facilitar a tarefa de medição e análise da *PA* (*Process Area*) de *MA* (*Measurement and Analysis*) do *CMMI*. A capacidade analítica e multidimensional, oferecida pelo componente *Analytic and Drill-down Capability based on Process/Responsibility*, foi desenvolvida para dar suporte aos gerentes intermediários e aos desenvolvedores de operações, sendo personalizada através de relatórios dinâmicos, exportações para planilhas *MS Excel* e funcionalidades de *drill-down/drill-up* similares aos recursos *OLAP*. O componente de Gestão e Controle de Qualidade (*Administration and Quality Control*) permite ao responsável por essas atividades definir novas medidas e permissões de acesso, e auditar a qualidade dos dados suportados pela ferramenta. A máquina analítica (*Analytical Engine - OLAP Technology*) oferece a funcionalidade de computar medições derivadas e agregadas, a partir das múltiplas dimensões do modelo analítico suportado pelo repositório. O repositório de medidas (*Measurement Repository*) é considerado o núcleo da ferramenta *MMR*. A sua base de dados é composta de entidades dos metadados, definidas por um metamodelo especificado por um diagrama de classe e associações, detalhado em [PAL03] *apud* [ABR03].

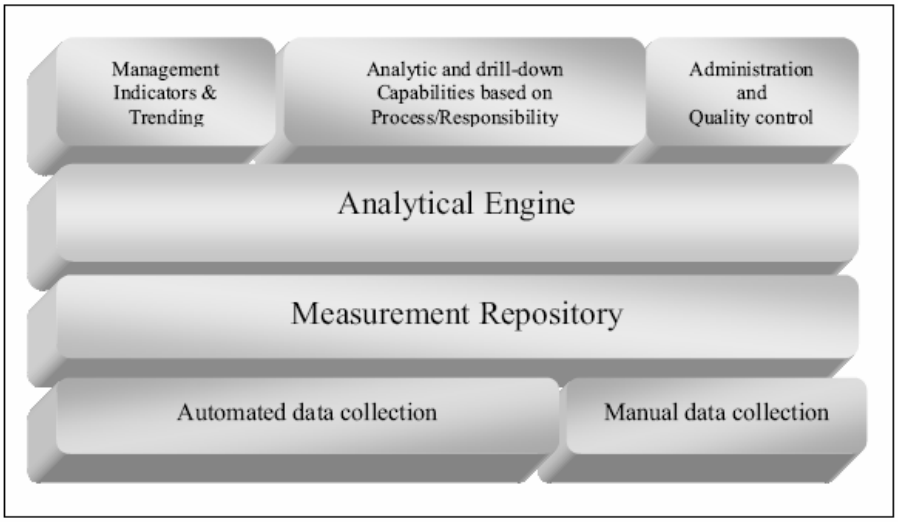


Figura 6 - Arquitetura de *Software* da *MMR* [PAL03].

4.1.1 Considerações Sobre a *MMR*

Com base na Tabela 2 percebe-se que a ferramenta *MMR* apresenta a freqüência e a latência do processo de coleta das métricas variando na ordem de dias, podendo chegar a meses. Para monitoração esses valores de tempo são considerados, respectivamente, pouco

aceitáveis e inaceitáveis. Para as demais atividades, podem ser aceitáveis. É importante salientar que esses valores de tempo são conseqüentes do processo manual e pouco freqüente de coleta das métricas. No que tange à heterogeneidade, pode-se destacar na *MMR*, apenas, a presença de um ambiente homogêneo ou heterogêneo de ferramentas, porém sem maiores detalhes de como os mesmos são suportados pela ferramenta. Por fim, com relação à intrusão conclui-se que o processo de coleta da *MMR* é bastante laborioso e manual, sendo considerado inaceitável para monitoração e pouco aceitável para análise e previsão. Além disso, essa ferramenta não esclarece como a flexibilidade provida pelo seu modelo analítico garante a consolidação das métricas, dos diferentes projetos, e a manutenção da sua consistência ao longo da sua evolução das métricas.

Tabela 2 – Tabela de interações tratadas pela MMR.

	Atividades de Supervisão				
Nível de Mensuração	**Análise**	**Monitoração**	**Previsão**	**Plataforma Computacional**	
De Projeto Organizacional	Dias (√) Meses (±)	Dias (±) Meses (×)	Dias (√) Meses (±)	**Latência**	
De Projeto Organizacional	Dias (√) Meses (±)	Dias (±) Meses (×)	Dias (√) Meses (±)	**Freqüência**	
De Projeto	-	-	-	**Modelos de Processo**	**Heterogeneidade**
	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	**Ferramentas**	
	-	-	-	**Projetos**	
	-	-	-	**Nível de Isolamento**	
Organizacional	-	-	-	**Modelos de Processo**	
	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	**Ferramentas**	
	-	-	-	**Projetos**	
	-	-	-	**Nível de Isolamento**	
De Projeto Organizacional	Alto (±)	Alto (×)	Alto (±)	**Intrusão**	

4.2 *BPI* e *iBOM*

BPI [CAS02a] consiste numa arquitetura geral, composta por um pacote de ferramentas, desenvolvida pela *Hewlett-Packard* (*HP*), para oferecer suporte, automatizado

ou semi-automatizado, à gestão de qualidade do processo de negócio, do *Business Process Management System* (*BPMS*). Esse pacote permite realizar atividades de:

- **Análise:** possibilita a realização de análises completas das execuções dos processos, tanto da perspectiva do negócio (*e.g.* alto nível, número de processos terminados com "sucesso"), como dos analistas de TI (*e.g.* baixo, média de tempo de execução por nó), apresentando funcionalidades para reportar dados, que auxiliam a identificação de prováveis causas de determinados comportamentos dos processos.
- **Previsão:** possibilita derivar modelos de previsão e aplicá-los nos processos em execução, para identificar exceções ou comportamentos indesejados.
- **Monitoração:** possibilita monitorar instâncias de processos em execução, informando sobre situações indesejadas ou incomuns. Dessa forma, os usuários podem verificar o status do sistema, dos processos, dos serviços e dos recursos. Fora isso, também, podem ser definidas situações críticas e alertas, quando as mesmas forem detectadas.

Dessa forma a solução *BPI* pretende: (i) identificar a arquitetura e as tecnologias que podem proporcionar as atividades mencionadas; (ii) permitir a definição de conceitos e métricas que permitem estabelecer níveis de negócio e análises quantitativas dos processos; (iii) desenvolver técnicas para facilitar a sua utilização, objetivando que a extração do conhecimento desejada, seja obtida sem nenhuma escrita de código; (iv) entender como realizar a interação com o *BPMS* e com os usuários, objetivando reportar e corrigir situações críticas em um intervalo de tempo adequado, para que ações corretivas possam ser efetuadas.

O BPI assume a execução de processos de negócio em um sistema de gerência de *workflow* específico denominado *HP Process Manager* (*HPPM*), gerando e armazenando dados em *logs*. A arquitetura do BPI (Figura 7) é responsável por estabelecer um ambiente para monitoramento em tempo real, análise, gerenciamento e otimização de processos de negócio. Essa arquitetura apresenta três componentes principais: *PDW Loader*, *Process Mining Engine* e *Cockpit*.

Primeiramente, O *PDW Loarder* extrai os dados dos *logs* de processos (*Process A..B Audit Logs*), verificando a sua consistência, calculando as métricas do negócio e inserindo os dados no *PDW*. Esse consiste em um *DW* e está estruturado segundo um modelo analítico que possibilita a definição e a rápida execução de relatórios agregados e detalhados, além de um vasto conjunto de funcionalidades analíticas. O modelo analítico do *PDW* apresenta estrutura multidimensional, conforme um modelo estrela, ilustrado pela Figura 8, onde o *Process State*

Changes e o *Service State Changes* são as tabelas fato e as demais constituem as dimensões. Com base nessa estrutura, os usuários podem, por exemplo, examinar quantas instâncias de processos são inicializadas por um determinado usuário dentro de um intervalo de tempo específico. O *PDW Loader* captura os dados dos *logs* e insere-os no *PDW*, podendo ser ativado periodicamente (agendado) ou sob requisição. Durante a carga, o *loader* verifica a consistência dos dados dos *logs* e corrige informações errôneas, que poderiam tornar as análises mais complexas ou tendendo a erros.

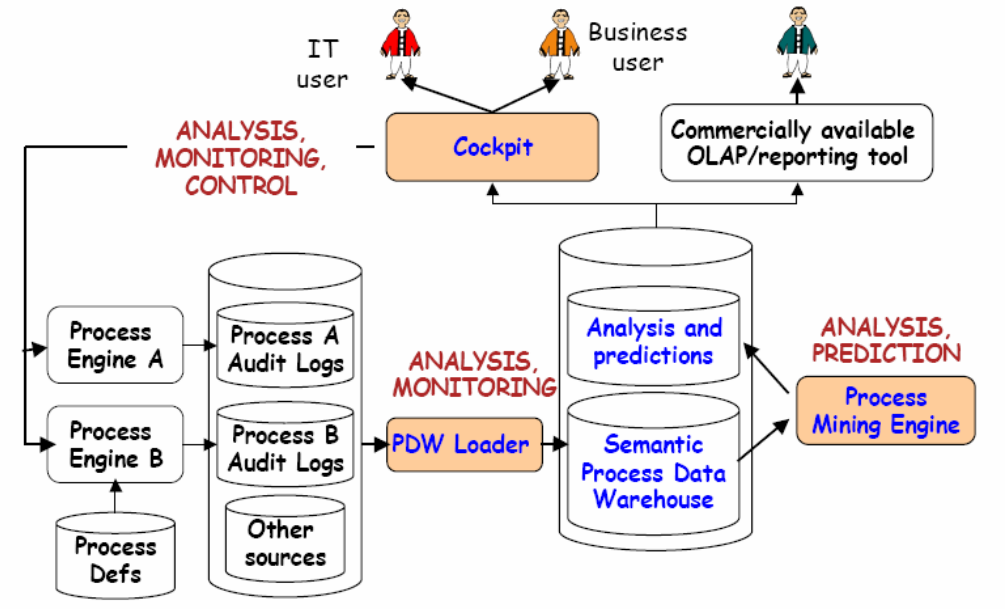


Figura 7 - Arquitetura do pacote de aplicações BPI [CAS02a].

Após o armazenamento dos dados no *DW*, o *Process Mining Engine* aplica técnicas de mineração de dados, buscando produzir modelos úteis para predizer a ocorrência de determinados comportamentos dos processos em execução, bem como identificar as causas desses comportamentos. Por fim, as informações são apresentadas pela interface *Cockpit*, projetada para produzir relatórios aos usuários do negócio. A mesma apresenta os dados do *DW* de maneira simplificada, permitindo que os analistas definam uma série de consultas de maneira intuitiva, sem exigir edição de código. Além disso, essa interface apresenta análises de processos e monitoração dos mesmos através de métricas e indicadores, apresentados na forma de gráficos e *dashboards*.

O *iBOM* [CAS05] é considerado uma evolução do *BPI* e inclui a funcionalidade de lidar com fontes de dados heterogêneas, por intermédio de um componente especial, chamado *Abstract Process Monitor* (*APM*), e permite a utilização de métricas definidas pelos usuários. O *APM* é basicamente um componente para coletar e interpretar métricas, no contexto de

processos abstratos. Assim, é possível realizar previsões, simulações e otimizações dos processos a partir do momento em que os dados dos processos estão disponíveis, e os valores das métricas estão computados e analisados. Os componentes responsáveis por essas funcionalidades obtêm esses dados e os valores das métricas do *APM*.

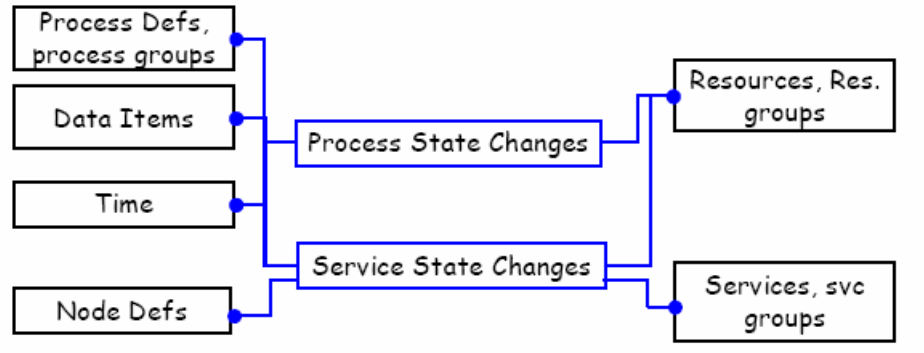


Figura 8 - Visão geral do esquema do *PDW* [CAS02a].

4.2.1 Considerações Sobre o *BPI*/*iBOM*

A partir dos domínios apresentados na Tabela 3, referentes ao *BPI* e ao *iBOM*, é possível perceber que ambos oferecem suporte às atividades de análise, previsão e otimização, apresentando tanto para a latência, quanto à freqüência, um domínio de tempo na ordem de horas, podendo em alguns casos aproximar-se do processamento em tempo-real. Além disso, apenas, o *iBOM* permite que métricas armazenadas em diferentes fontes de dados originais possam ser consolidadas, por intermédio do componente *APM*, oferecendo suporte a um ambiente heterogêneo de ferramentas. Essas soluções não discutem explicitamente o nível de isolamento. Porém, como o *APM* lida com diferentes ferramentas, é possível inferir que não existem dificuldades de acesso aos dados e as suas respectivas fontes originais. Por fim, devido às funcionalidades oferecidas pelo *PDW Loader* é possível concluir que as soluções são isentas de intrusão, caracterizando um processo de ETC completamente automatizado. Vale ressaltar que as rotinas de extração e os métodos de transformação não são relatados na descrição da solução. Contudo, *BPI*/*iBOM* endereça processos de negócio apoiados por uma ferramenta de automação de *workflow*. Com efeito, é considerado que os valores das métricas estão sempre computacionalmente disponíveis e, portanto, acessíveis por alguma ferramenta,

o que implica que os recursos não têm de apropriar nenhum valor de métrica. Essa situação inexiste em um ambiente de desenvolvimento de *software.*

Tabela 3 – Tabela de interações tratadas pela *BPI*/*iBOM*.

	Atividades de Supervisão				
Nível de Mensuração	**Análise**	**Monitoração**	**Previsão**	**Plataforma Computacional**	
De Projeto Organizacional	Horas (√)	Horas (√)	Horas (√)	**Latência**	
De Projeto Organizacional	Horas (√)	Horas (√)	Horas (√)	**Freqüência**	
De Projeto	-	-	-	**Modelos de Processo**	**Heterogeneidade**
	Heterogêneo (S)	Heterogêneo (S)	Heterogêneo (S)	**Ferramentas**	
	-	-	-	**Projetos**	
	Compartilhado(S)	Compartilhado(S)	Compartilhado(S)	**Nível de Isolamento**	
Organizacional	-	-	-	**Modelos de Processo**	
	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	**Ferramentas**	
	-	-	-	**Projetos**	
	Compartilhado(S)	Compartilhado(S)	Compartilhado(S)	**Nível de Isolamento**	
De Projeto Organizacional	Isento (√)	Isento (√)	Isento (√)	**Intrusão**	

4.3 *SPDW*

O *SPDW* [BEC06] é um ambiente de *Data Warehousing* para apoiar o Programa de Métricas da *HP EAS* Brasil, desenvolvido pelo projeto de parceria entre o Programa de Pós-Graduação de Ciência da Computação da PUCRS (PPGCC-PUCRS) e a *HP EAS* Brasil, durante o seu processo de certificação *CMM3*.

A Figura 9 ilustra a arquitetura do *SPDW*, organizada em camadas distintas: Camada de Integração das Aplicações (*Application Integration Component*), Camada de Integração dos Dados (*Data Integration Component*) e Camada dos Componentes de Apresentação (*Presentation Components*).

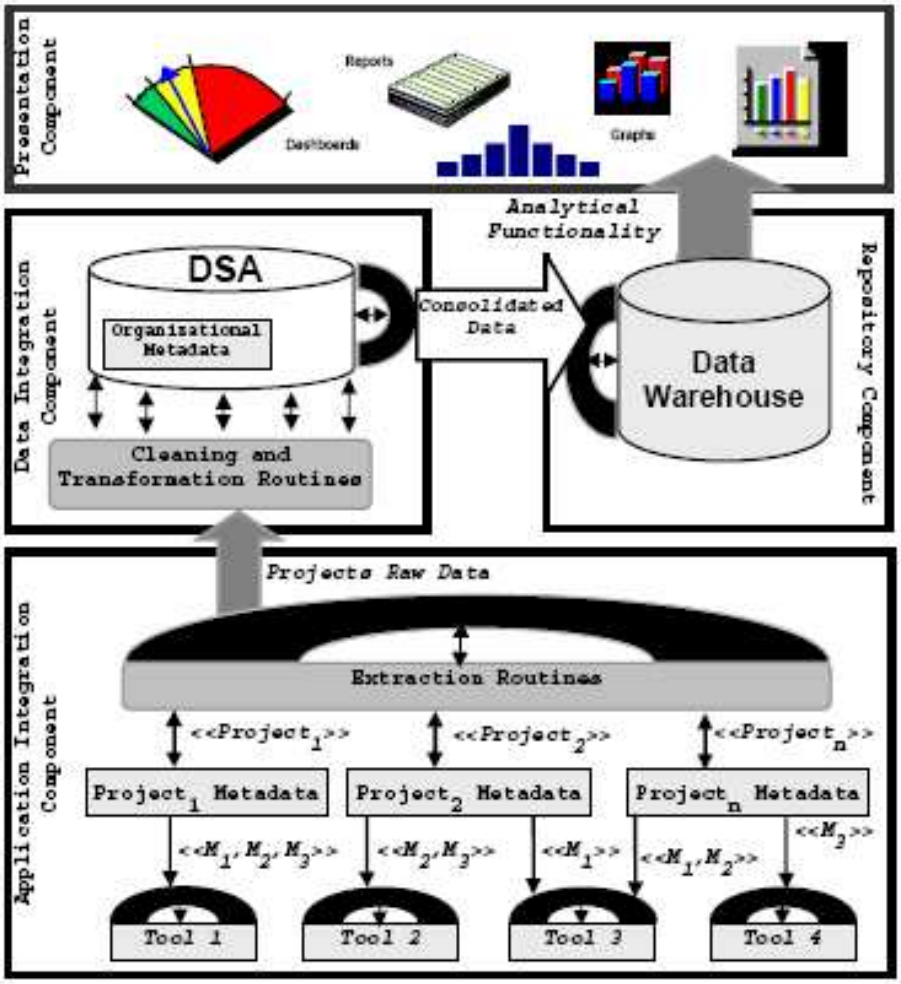


Figura 9 - Arquitetura do *SPDW* [BEC06].

A Camada de Integração de Aplicações é responsável pela extração automática dos dados diretamente das fontes dos projetos, advindos de diferentes ferramentas (*Tools*) de apoio ao PDS. Os dados são carregados em uma *DSA*, que pertence à Camada de Integração de Dados, onde tais dados são devidamente transformados conforme o padrão organizacional e, posteriormente, armazenados no *DW*. Essa extração é feita com o uso de *WS*, representados por semicírculos pretos, baseados nos metadados dos projetos (*Project Metadata*). Por último, a Camada de Componentes de Apresentação permite a exibição das métricas para análise armazenadas no repositório (*DW*), de acordo com os diferentes perfis de usuário (organizacional, de projeto, de gerência e técnico) e objetivos de análise. Essa camada é composta por dois mecanismos de apresentação: recursos *OLAP* e *dashboards*.

A arquitetura do *SPDW* apresenta apenas um fluxo unidirecional de dados, tendo a *Extraction Routine* como coordenadora do processo de extração das métricas. Dessa forma, somente essa última consegue realizar a comunicação entre as camadas de Integração de Aplicações e de Dados, impedindo que os *wrappers* ou outros serviços possam iniciar o

processo de ETC das métricas, de maneira pró-ativa quando novos lançamentos forem detectados ou quando o gestor do projeto achar oportuno.

Esse ambiente foi positivamente apreciado durante o processo de avaliação de qualidade e a sua adoção trouxe resultados benéficos e significativos à organização, associados com a padronização, regularidade e consistência de captura das métricas. Além disso, o *SPDW* melhorou os recursos analíticos e de comunicação, devido à utilização de indicadores de qualidade (*dashboards*), relatórios e análises pré-definidos, e recursos *OLAP*. Esses últimos permitem que o grupo de qualidade e os responsáveis pelos projetos possam explorar o *DW* para encontrar as razões das deficiências ou as melhorias detectadas, conseqüentes das ações tomadas. A parceira também utiliza o *SPDW* para solidificar sua credibilidade, apresentando os indicadores de qualidade dos processos, dos projetos, dos produtos e da satisfação dos clientes, durante as apresentações de pré-vendas.

As seções a seguir detalham, do *SPDW*, o programa de métricas adotado, o modelo analítico desenvolvido e o processo de ETC especificado. Este detalhamento se faz necessário, pois a pesquisa sendo relatada nesta dissertação tem como ponto de partida as contribuições oferecidas pelo *SPDW*.

4.3.1 Programa de Métricas

Para atender as exigências de *PA* de *MA*, do *CMM2*, foi estabelecido um Programa de Métricas Organizacional (Tabela 4), onde as métricas encontram-se agrupadas em áreas de qualidade (*Quality Areas*) distintas: *Schedule* (Tempo, Cronograma), *Effort* (Esforço), *Size* (Tamanho), *Cost* (Custo), *Requirements* (Requisitos) e *Quality* (Qualidade, Defeitos). Cabe salientar que essas métricas permitem apenas a execução de análises e predições do PDS, não sendo adequadas à monitoração, pois não seguem nenhuma das técnicas usadas para tanto (*e.g. EVA* detalhada na Seção 2.3)

Esse conjunto de métricas não permite acompanhar regularmente o andamento do planejamento do projeto, pois detecta desvios significativos somente quando o tempo estimado para executar uma determinada fase, iteração ou versão está encerrado. Dessa maneira, não é possível antecipar decisões e tomar ações corretivas para evitar que esses desvios interfiram no prazo ou no custo do produto final. Por exemplo, a métrica Variação de *Baseline* Original (***SVOB*** *– Schedule Variance Original Baseline*) quando avaliada, segundo índices de desempenho (desejados, pouco desejados ou indesejados) definidos por valores de mercado (*e.g. GDIC IQMS* [HPC07c] e *PMBOK - Risk Management* (*Moderate Impact*)

[PMI04] apresenta valor indesejado somente no momento em que foi concluída ou após ultrapassar o limite de prazo definido para a sua execução. Por essa razão, a mesma não reflete a situação real do andamento do projeto, não sendo úteis para monitoração.

Tabela 4 - Programa de Métricas do *SPDW* [BEC06].

Quality Areas	*Derived Metrics*	*Direct Metrics*
Schedule	***SVOB*** *– Schedule Variance Original Baseline* ***SVRB*** *– Schedule Variance Revised Baseline*	***ASD*** *– Actual Start Date* ***AED*** *– Actual End Date* ***OBSD*** *– Original Baseline Start Date* ***OBED*** *– Original Baseline End Date* ***RBSD*** *– Revised Baseline Start Date* ***RBED*** *– Revised Baseline End Date*
Effort	***EVOB*** *– Effort Variance Original Baseline* ***EVRB*** *– Effort Variance Revised Baseline* ***PR*** *– Productivity*	***AE*** *– Actual Effort* ***OBE*** *– Original Baseline Effort* ***RBE*** *– Revised Baseline Effort* ***AS*** *– Actual Size*
Size	***SZVOB*** *– Size Variance Original Baseline* ***SZVRB*** *– Size Variance Revised Baseline*	***OBS*** *– Original Baseline Size* ***RBS*** *– Revised Baseline Size* ***AS*** *– Actual Size*
Cost	***CV*** *– Cost Variance* ***CQ*** *– Cost of Quality*	***AC*** *– Actual Cost* ***OBC*** *– Original Baseline Cost* ***RBC*** *– Revised Baseline Cost* ***ACAR*** *- Actual Cost of Activity Revision* ***ACTP*** *– Actual Cost of Test Phase* ***ACAQ*** *– Actual Cost of Activity Quality* ***ACRWA*** *– Actual Cost of Rework Activity*
Requirements	***RV*** *– Requirements Volatility*	***NACR*** *– No. of Approved Change Requests* ***NAR*** *– No. of Added Requirements* ***NDR*** *– No. of Deleted Requirements* ***NMR*** *– No. of Modified Requirements*
Quality	***DRE*** *– Defect Removal Efficiency* ***DDD*** *– Delivered Defect Density* ***IDD*** *– Internal Defect Density* ***RE*** *– Review Efficiency* ***CS*** *– Customer Satisfaction*	***NID*** *– No. of Defects Internally Found* ***NED*** *– No. of Defects Found by Clients* ***AS*** *– Actual Size* ***CSI*** *– Customer Satisfaction Index*

4.3.2 *Data Warehouse* (*DW*)

O repositório de dados, estruturado na forma de um *DW*, é o núcleo da arquitetura do *SPDW*. Nele, os dados dos projetos são armazenados para dar suporte à análise do PDS, de acordo com as métricas definidas pela organização.

A Figura 10 mostra a estrutura estendida dos projetos suportados pelo DW, por intermédio de um diagrama de classes *UML*. A mesma permite a adoção de projetos com ciclo de vida cascata e iterativo. Dessa forma, cada projeto (Project) pode apresentar uma ou mais versões (Release), sendo que as mesmas podem conter um conjunto de iterações (Iteration) ou fases (Phase). Porém, uma fase pode ou não pertencer a uma iteração. Essa condição somente é possível devido à presença da superclasse estágio (Stage). Os atributos das classes do diagrama correspondem às métricas da Tabela 4, representando os diferentes níveis de detalhe em que as mesmas podem ser capturadas, ou produzidas, para cada especialização do *OSSP* (*Organization´s Standard Software Process*). Estimativas de Esforço e Tempo são estabelecidas em termos de iteração e fase, enquanto custo e tamanho são estimados no nível de versão. Os valores reais dessas métricas são capturados no grão de atividade. Já os defeitos são mensurados a partir dos estágios. Os requisitos e a satisfação do cliente são medidos no nível de versão. Não existem estimativas para as áreas de requisitos e qualidade (defeitos e satisfação do cliente).

Durante a concepção do *SPDW* foi imposto pela organização que os projetos deveriam classificar as suas atividades em: trabalho, retrabalho, revisão e qualidade. Além disso, também foi constatado que as estimativas de esforço não eram realizadas até o grão de atividades e, por essa razão esse nível apresenta apenas métricas com valores reais (efetivamente realizados). Essas decisões foram cruciais na definição da granularidade das tabelas fato que compõem o modelo analítico.

A Figura 11 Apresenta o modelo analítico do *SPDW*. Este é constituído por uma estrutura dimensional organizada na forma de uma constelação de fatos, onde as tabelas dimensão representam as perspectivas de análise dos projetos e as tabelas fato armazenam as métricas nos seus diferentes níveis, considerando as diferentes granularidades de informação. É importante salientar que esse modelo é totalmente dependente do programa de métricas e dos processos organizacionais (*OSSP* e *PDP*).

Segundo [BEC06], o modelo analítico considera um conjunto restrito, porém significante de métricas. Além disso, sua estrutura multidimensional favorece a inclusão de novas métricas, por intermédio da simples adição de dimensões [KIM98], desde que sua

granularidade esteja de acordo com o modelo. Nesse sentido, métricas com granularidade diferente devem ser incluídas a partir da criação de novas tabelas fato. Essa facilidade agrega flexibilidade ao modelo, conseqüentemente também favorece a extensão do programa de métricas e a evolução dos processos adotados pela organização.

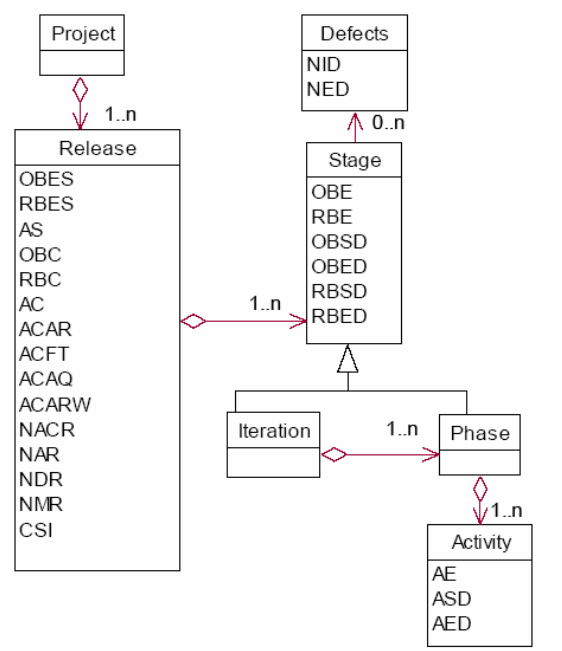


Figura 10 - Estrutura dos Projetos [BEC06].

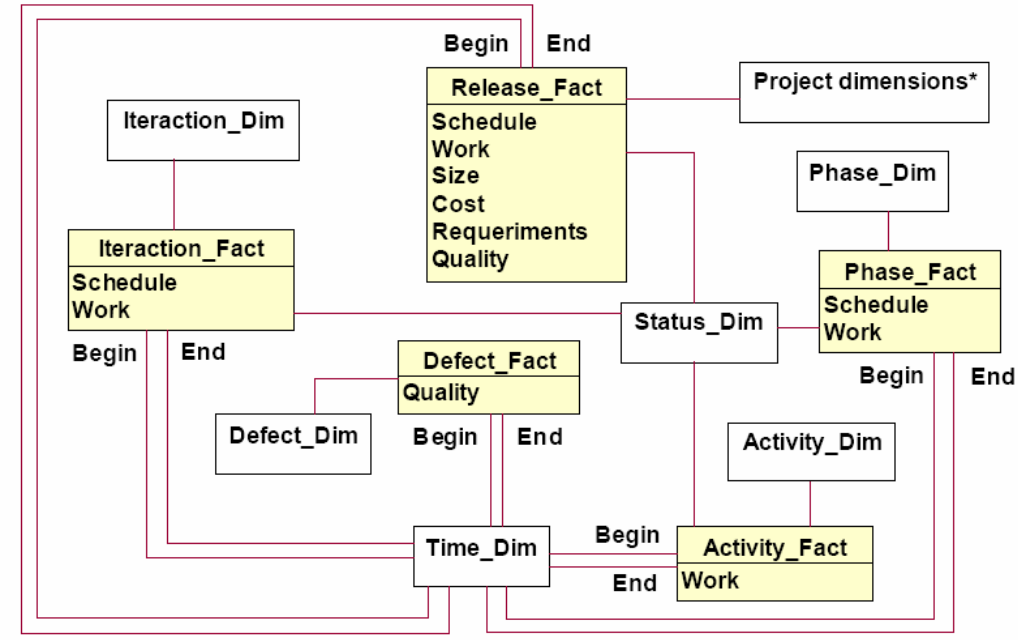


Figura 11 - Modelo Analítico *SPDW* [BEC06].

O modelo analítico do *SPDW* foi desenvolvido para análise e apresenta limitações quanto à granularidade das informações e às perspectivas de análise. A partir do mesmo não é possível acompanhar regularmente o desempenho das atividades de maneira individual, ou sumarizá-las por intervalo de data, devido à consolidação estabelecida por tipo atividade. Além disso, as informações de custo são apresentadas, apenas, no nível de versões impedindo, por exemplo, a verificação do valor consumido por determinada fase ou iteração, bem como a monitoração dos mesmos em uma granularidade menor. Outra limitação do modelo original está relacionada com a existência de métricas não aditivas nas tabelas fato. As mesmas impedem que sejam realizadas sumarizações das métricas nos níveis mais altos (versão, iteração ou fase) a partir do *Activity_Fact* (Fato_Atividade). Por essa razão, o original apresenta uma tabela fato para cada nível hierárquico da estrutura do cronograma: Atividade, Fase e Iteração.

4.3.3 Processo de ETC

Na perspectiva de captura dos dados, as diferentes particularidades existentes nos projetos implicam em lidar com os aspectos envolvidos pelas três dimensões ortogonais (Seção 3.1). Por essa razão, o processo de ETC de métricas de *software* pode ser considerado complexo. Além disso, no contexto do *SPDW*, esse processo também envolve a interação entre componentes de camadas diferentes, Integração das Aplicações e Integração dos Dados, para carregar os dados de maneira concisa e consolidada no *DW*.

O ETC de métricas de *software*, proposto pelo *SPDW*, trata a liberdade dos projetos de escolher diferentes ferramentas, que variam desde simples planilhas *MS Excel* a ferramentas dedicadas, como *MS Project* e *IBM Rational ClearQuest*, entre outras. O mesmo também possibilita que cada um dos projetos adote processos especializados do *OSSP* (*PDP*), que introduzem variações quanto ao ciclo de vida (cascata, iterativo) e modelos de gerenciamento (orientado a fases, orientado a entregáveis) dos projetos. Para tratar essas questões de heterogeneidade, o processo de ETC segue um padrão arquitetural orientado a serviços, responsável pela comunicação entre os diferentes componentes da arquitetura.

Na Figura 9, onde a arquitetura do *SPDW* está ilustrada, os serviços (*WS*) estão representados por semicírculos pretos, que simbolizam os *wrappers* das ferramentas, os quais utilizam mensagens *SOAP* para realizar a comunicação entre os diferentes componentes da arquitetura, encapsulando as especificidades das ferramentas (*Tools*) utilizadas pelos projetos. No entanto, para que a AOS possa ser considerada completa, essa solução deveria apresentar

a definição dos serviços que compõem o processo automatizado de ETC das métricas. Além disso, vale ressaltar que o *SPDW* não apresenta uma especificação detalhada ou uma prototipação das Rotinas de Extração e das de Transformação, nem dos *wrappers*. Nesse trabalho apenas são realizadas descrições superficiais desses componentes, mencionando o papel de cada um deles na arquitetura. Ainda, cabe mencionar que o *SPDW* não discute a freqüência e a latência de extração das métricas, muito menos as questões relacionadas com a carga incremental.

Com relação à Rotina de Extração (*Extraction Routine)*, a mesma tem por objetivo localizar os *wrappers* das ferramentas e repassar as informações, contidas nos metadados dos projetos, para que eles possam localizar o dado requerido na base de dados das ferramentas. Nessa arquitetura, são mencionados dois tipos de metadados. O primeiro, Metadado do Projeto (*Project Metadata*), contém informações para o mapeamento entre o dado requerido e o armazenado na fonte, enquanto que o segundo, Metadado da Organização (*Organizational Metadata*), contém informações necessárias às rotinas de limpeza e transformação da Camada de Integração dos dados. Ambos são definidos por esquemas *XML* e contribuem para a flexibilidade do processo de ETC das métricas de *software*.

A automatização do ETC das métricas, por intermédio da utilização de *WS* e metadados, possibilita que as mesmas possam ser capturadas, transformadas e consolidadas no repositório, com baixa intrusão e alta flexibilidade, em um ambiente heterogêneo. Assim, a automatização facilita a evolução do programa de métricas organizacional e a utilização de novas ferramentas, caracterizando a arquitetura proposta, como sendo uma solução com baixo acoplamento entre as ferramentas e alta coesão dos dados.

4.3.4 Considerações Sobre o *SPDW*

A partir da Tabela 5 percebe-se que o *SPDW* não oferece suporte à monitoração e à previsão, tratando apenas os aspectos relacionados com as atividades de análise. Nesse sentido, esse ambiente apresenta uma baixa freqüência e alta latência, conseqüentes de um processo semi-automatizado de ETC, variando na ordem de dias e meses. Ambas são consideradas, respectivamente, aceitáveis e pouco aceitáveis para analisar a qualidade de *software*. Com relação à heterogeneidade, nota-se que o *SPDW* descreve como lidar com todos os aspectos mencionados. Por fim, também é possível concluir que o seu processo de ETC apresenta um baixo nível de intrusão.

Tabela 5 – Tabela de interações tratadas pelo ambiente do *SPDW*.

	Atividades de Supervisão				
Nível de Mensuração	**Análise**	**Monitoração**	**Previsão**	**Plataforma Computacional**	
De Projeto Organizacional	Dias (√) Meses (±)	-	-	**Latência**	
De Projeto Organizacional	Dias (√) Meses (±)	-	-	**Freqüência**	
De Projeto	Homogêneo (S)	-	-	**Modelos de Processo**	**Heterogeneidade**
	Homogêneo (S) Heterogêneo (S)	-	-	**Ferramentas**	
	Pequeno (Ñ) Grande (Ñ)	-	-	**Projetos**	
	Compartilhado (S)	-	-	**Nível de Isolamento**	
Organizacional	Homogêneo (S) Heterogêneo (S)	-	-	**Modelos de Processo**	
	Homogêneo (S) Heterogêneo (S)	-	-	**Ferramentas**	
	Pequeno (S) Grande (S)	-	-	**Projetos**	
	Compartilhado (S)	-	-	**Nível de Isolamento**	
De Projeto Organizacional	Baixo (√)	-	-	**Intrusão**	

É importante mencionar que o *SPDW* apresenta uma arquitetura em alto nível, sem detalhar as especificidades dos seus componentes, a latência, a freqüência e a carga incremental, considerados aspectos essenciais para armazenar as métricas no Repositório de Dados. Além disso, o seu modelo analítico contém problemas de construção, devido à existência de métricas não aditivas nos seus fatos, impedindo a sua sumarização nos níveis superiores e exigindo a utilização de um número maior de tabelas fato. Com relação aos testes feitos à época, foram implementados apenas protótipos de alguns *wrappers* de extração. Essa solução também não menciona como a Rotina de Extração inicia o processo de ETC nem demonstra como a baixa intrusão é garantida.

Atualmente, no ambiente da *HP EAS* Brasil, encontra-se em funcionamento um processo semi-automatizado de ETC, com alto nível de intrusão, baseado no proposto pelo *SPDW*, desenvolvido por uma outra empresa de TI. Logo, conclui-se que o *SPDW* propõe uma solução em alto nível, sem ter especificado e testado suficientemente as soluções computacionais adotadas para resolver os aspectos críticos destacados na Tabela 1.

4.4 Considerações sobre os Trabalhos Relacionados

Esse capítulo apresentou os seguintes trabalhos: 1) *MMR*, 2) *BPI*/*iBOM* e 3) *SPDW*, vistos a partir das três dimensões, definidas no capítulo anterior. A Tabela 6 contém uma consolidação apenas dos domínios dos aspectos críticos (Tabela 1, Seção 3.1.3.2) e referências aos trabalhos estudados, conforme a legenda abaixo da tabela. Por questões de legibilidade, quando o aspecto crítico não foi atendido por nenhum dos trabalhos, o mesmo foi marcado com "X".

A partir da leitura da Tabela 6 percebe-se que a proposta do conjunto de ferramentas *BPI*/*iBOM* consegue solucionar uma grande parte dos aspectos relacionados com à latência, à freqüência, à heterogeneidade de ferramentas, apresentando um nível isento de intrusão e considerando as três dimensões. É importante salientar que essas soluções são aplicadas em um ambiente de execução de processos de negócio, sem considerar as particularidades do ambiente de desenvolvimento de *software* (*e.g.* latência inerente). Por essa razão, os aspectos mencionados são parcialmente atendidos.

Por sua vez a *MMR* atende quase os mesmos aspectos que a solução *BPI*/*iBOM*, porém considerando um ambiente de desenvolvimento de *software*. Contudo, essa ferramenta apresenta um processo de coleta de métricas eminentemente manual, exigindo um lançamento individual dos valores por intermédio de uma interface específica. Dessa forma, essa solução não atende o aspecto referente à baixa intrusão, considerado fundamental para mensurar a qualidade de *software*, freqüentemente e com uma latência na ordem de horas.

Já o *SPDW*, atende somente a dimensão de atividade de análise, tanto para o nível de projeto quanto organizacional, considerando intrusão e todos os aspectos críticos relacionados com heterogeneidade, com exceção do nível de isolamento. Diferentemente dos outros trabalhos é o único que trata modelos de processo e projetos. Porém, não oferece suporte à monitoração, essencial para permitir um acompanhamento regular do planejamento inicial e crucial para atingir os níveis mais altos dos modelos de qualidade [SOF07] [SEI06].

É importante destacar que, no cenário das três dimensões, a monitoração no nível organizacional, considerando todos os aspectos da plataforma computacional, apresenta os domínios mais críticos a serem atendidos por uma solução ideal (Seção 3.2). Nesse sentido, a partir da Tabela 6, é possível constatar que os trabalhos relacionados não tratam adequadamente todos os aspectos relevantes desse cenário. Dessa forma, esse capítulo buscou destacar os requisitos da solução que não foram atendidos.

Tabela 6 – Tabela de interações tratadas pelos Trabalhos Relacionados.

	Atividades de Supervisão				
Nível de Mensuração	**Análise**	**Monitoração**	**Previsão**	**Plataforma Computacional**	
De Projeto Organizacional	Horas (√) 1, 2 Dias (√) 1, 2, 3	Horas (√) 1, 2	Horas (√) 1, 2 Dias (√) 1, 2	**Latência**	
De Projeto Organizacional	Horas (√) 1, 2 Dias (√) 1, 2, 3	Horas (√) 1, 2	Horas (√) 1, 2 Dias (√) 1, 2	**Freqüência**	
De Projeto				**Modelos de Processo**	**Heterogeneidade**
	Heterogêneo (S) 1, 2, 3	Heterogêneo (S) 1, 2	Heterogêneo (S) 1, 2	**Ferramentas**	
				Projetos	
				Nível de Isolamento	
Organizacional	Heterogêneo (S) 3	Heterogêneo (S) X	Heterogêneo (S) X	**Modelos de Processo**	
	Heterogêneo (S) 1, 2, 3	Heterogêneo (S) 1, 2	Heterogêneo (S) 1, 2	**Ferramentas**	
	Pequeno (S) 3 Grande (S) 3	Pequeno (S) X Grande (S) X	Pequeno (S) X Grande (S) X	**Projetos**	
	Isolado (S) X	Isolado (S) X	Isolado (S) X	**Nível de Isolamento**	
De Projeto Organizacional	Isento (√) 2 Baixo (√) 2, 3 Médio (√) 2, 3	Isento (√) 2 Baixo (√) 2	Isento (√) 2 Baixo (√) 2 Médio (√) 2	**Intrusão**	

Legenda: 1 – *MMR*, 2 – *BPI*/*iBOM*, 3 - *SPDW*

5 ESTUDO DE CASO

Neste capítulo é apresentado o cenário real da operação de *software* parceira, certificada *CMM3*, e considerada de grande porte na América Latina. Para tanto, primeiramente é feita uma descrição desta operação, relatando a solução de automação computacional adotada, denominada Base Organizacional (BO) e seus principais componentes. Posteriormente, são apontadas algumas dificuldades encontradas neste cenário. Por fim, é apresentada uma estimativa do esforço necessário para tornar a solução atual mais automatizada e menos intrusiva.

5.1 Base Organizacional

A operação de *software* escolhida apresenta como solução, para mensurar a sua qualidade, uma adaptação do ambiente proposto pelo *SPDW*, denominada Base Organizacional (BO). Essa adoção parcial foi conseqüente da opção, por parte da parceira, em implantar rapidamente a BO, o que foi feito licenciando e adaptando uma Ferramenta de BI (*Business Intelligence*) de uma empresa de TI. A concepção da BO teve como meta principal prover uma visão unificada dos diferentes projetos da operação, de forma a viabilizar a comparação e análise dos mesmos, de maneira sistemática e confiável. Nesse sentido, foi desenvolvida uma arquitetura composta por três grandes componentes: a) Processo de Carga; b) Repositório; e c) Interface de *BI*, descritos sucintamente nas próximas seções.

5.1.1 Processo de Carga

O processo de Carga da BO é responsável por coletar métricas das seis áreas de qualidade (Esforço, Qualidade, Custo, Cronograma, Tamanho e Requisitos), relatadas na Seção 4.3.1, e consolidá-las no repositório central, a partir de um ambiente heterogêneo de projetos e ferramentas. Nesse ambiente as métricas de Cronograma encontram-se armazenadas no *MS Project*. Por sua vez, as de Qualidade podem ser capturadas do *Bugzilla*, *ClearQuest* ou *Mantis*, dependendo do projeto. Já as métricas referentes ao esforço realizado podem estar armazenadas na base de dados do *MS Project*, localizada no ambiente do projeto, ou no Banco de Horas, desenvolvido por um dos projetos da operação. E, por fim, os requisitos e os tamanhos podem estar armazenados em documentos distintos não estruturados, variando conforme o projeto.

A definição do processo de carga, adotado pela BO, apresenta uma série de etapas manuais e semi-automatizadas, executadas por pessoas específicas. A Figura 12 ilustra a ordem seqüencial de execução dessas etapas.

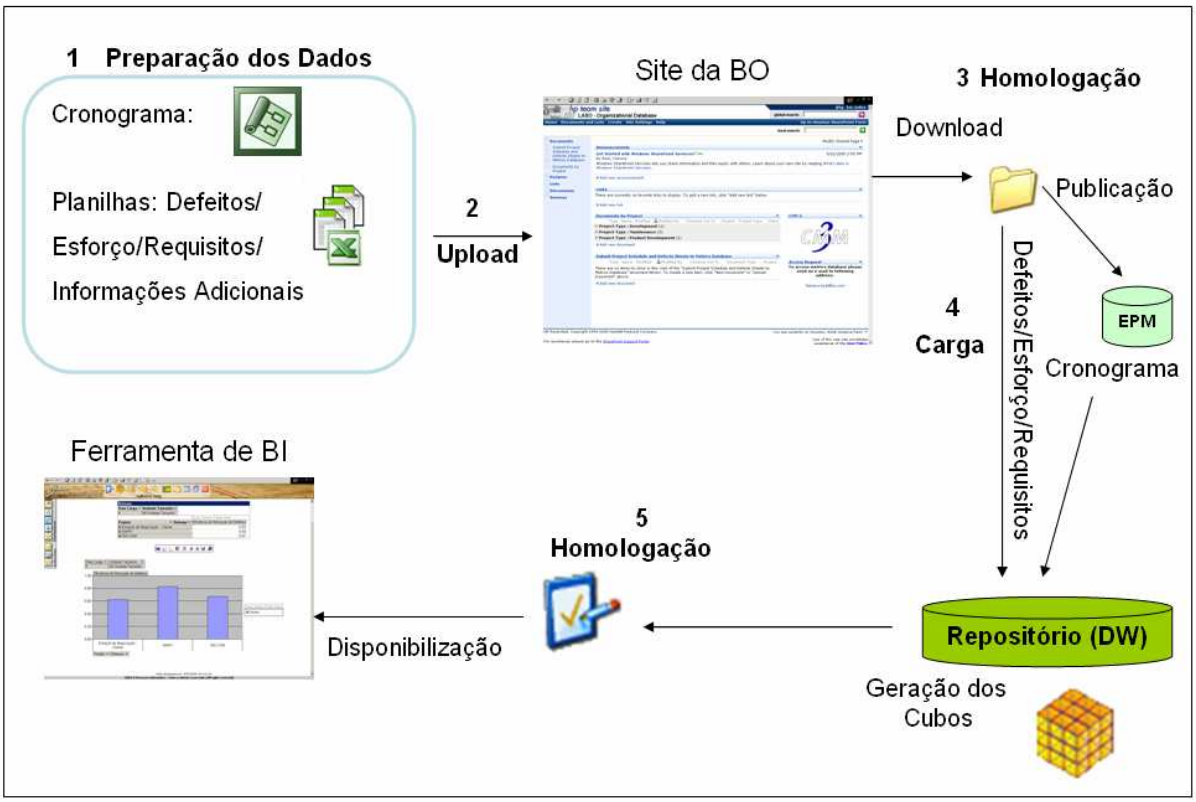


Figura 12 – Processo de Carga da BO.

1) Preparação dos dados: consiste na execução de uma série de atividades manuais, realizadas no ambiente de projeto, as quais exigem que um dos seus recursos seja deslocado das suas atividades regulares. Essas atividades têm por objetivo capturar as métricas de fontes distintas e organizá-las em arquivos estruturados, conforme um padrão determinado pela organização. Para tanto é necessário preparar um pacote contendo os seguintes arquivos: cronograma, esforço, defeito, requisito e informações adicionais.

O arquivo de cronograma possui valores referentes às estimativas das diferentes etapas do projeto, caracterizando iterações, fases e atividades. Esses dados estão dispostos na ferramenta *MS Project* na forma de itens, de acordo com uma ordem de classificação, que pode ser percebida visualmente conforme a disposição dos mesmos. Essa ordem é conseqüente do ciclo de vida (*e.g.* cascata, iterativo) ou do modelo de gestão (*e.g.* orientado a fases ou a entregáveis), adotado pelo projeto. Assim, a preparação desse arquivo requer a inclusão de três campos adicionais: nome da fase, tipo da atividade e número da iteração, informados de forma manual,

explicitando a classificação de cada dos seus itens. Além disso, é necessário realizar a exportação desses dados para um arquivo de extensão “.mpp”.

Os dados de esforço realizado devem ser exportados do Banco de Horas, conforme o padrão estabelecido [HPC05c]. Cabe salientar que, em um dos projetos, o esforço realizado encontra-se armazenado no cronograma. Por essa razão, não é necessário realizar consultas específicas para coletá-los, pois os mesmos são lançados pelos recursos diretamente no cronograma.

A preparação do arquivo de defeitos exige a execução de consultas nas bases de dados das ferramentas de *bugtracking* (*IBM Rational ClearQuest* e *Bugzilla*) e padronização dos seus resultados segundo a guia de registro de defeitos [HPC05d]. Com relação aos dados de requisitos, os mesmos não se encontram armazenados de forma estruturada e, por essa razão, não é possível realizar consultas para capturá-los. Portanto, os mesmos são coletados de forma manual e inseridos em uma planilha *MS Excel*, conforme o padrão documentado na guia de captura de requisitos [HPC05b].

O arquivo de informações adicionais apresenta dados de custo, tamanho da versão, unidade de tamanho (*e.g.* KLOC, PF, PMHP), tamanho da equipe, status (*e.g.* Fechando, Em andamento), entre outros. Os dados utilizados para alimentar esse arquivo são provenientes de diferentes fontes e o seu procedimento de construção varia conforme o projeto. No entanto, tanto a organização como os projetos não possuem uma documentação apropriada relatando os passos necessários para a obtenção dos valores mencionados.

2) *Upload* do Pacote de Carga: Concluída a fase de preparação dos dados, cada projeto gera um conjunto de arquivos, e esse é enviado por *upload* para o *site* da BO.

3) Homologação dos Dados Coletados: O responsável pela carga recebe um aviso que os arquivos se encontram à disposição. Assim, eles passam por um processo manual de conferência, onde ocorre a verificação de coerência geral do preenchimento e da formatação, o qual denomina-se homologação das fontes de informação, realizada segundo o documento *checklist* de carga [HPC05a]. Além disso, durante essa etapa também os dados que compõem o arquivo de informações adicionais são inseridos nos arquivos de cronograma, e esses últimos são publicados na base de dados organizacional do *EPM* (*Enterprise Project Manager*).

4) Realização da Carga: Depois que os dados são homologados, eles encontram-se no formato adequado para que possa ser realizado o processo de ETC das métricas. Esse é automatizado por intermédio de pacotes *DTS* (*Data Transformation Services*) do *SQL Server* 2000 [SQL07], tendo como pré-condição que todos os dados dos projetos estejam coletados,

transformados e disponibilizados, segundo o padrão organizacional especificado nos documentos de carga. Concluída essa etapa, os dados estão devidamente armazenados no repositório de dados da BO.

5) Homologação dos Dados apresentados pela Interface de *BI*: Quando a etapa anterior é concluída, o gerente de *Software* do projeto é notificado. Então, um novo processo de homologação de dados deve ser realizado, onde ocorre a conferência dos dados apresentados pela Interface de *BI*. Após a finalização dessa parte do processo de carga, uma notificação de homologação do conteúdo é inserida na interface.

5.1.2 Repositório

O repositório de métricas está estruturado na forma de um *DW* e contém um conjunto de tabelas, organizadas segundo um esquema dimensional do tipo constelação de fatos. As tabelas fato contêm medidas básicas e calculadas sobre versões, fases, iterações, tipos de atividades e defeitos. As tabelas dimensão contêm atributos que qualificam estes fatos, tais como: diferentes propriedades dos projetos, versões, iterações, fases, etc. Cabe salientar que o modelo analítico do repositório baseia-se na solução proposta pelo *SPDW* (Seção 4.3). Porém, foram feitas algumas adaptações devido às limitações da ferramenta que suporta a interface de *BI*. Maiores detalhes desse modelo e do conteúdo das suas tabelas podem ser encontrados em [CUN05].

5.1.3 Interface de *BI*

A interface de *BI* (Figura 13 e 14) da ferramenta licenciada consiste em um portal *web*, formado por: (i) recursos *OLAP* de acesso aos cubos de dados; (ii) um painel de indicadores; (iii) recursos para visualização das consultas realizadas pelos usuários; (iv) calendário e integração com correio eletrônico; etc. A interface *OLAP* permite apresentação dos dados através de tabelas dinâmicas, manipuladas com o apoio de um *browser*, facilidades de tabelas pivotantes e recursos tradicionais *OLAP*. A partir dos dados das tabelas pivotantes, gráficos podem ser gerados e visões sobre os cubos podem ser pré-definidas.

5.2 Dificuldades Encontradas no ambiente da Base Organizacional

Esta seção resume as principais dificuldades encontradas no ambiente da BO. Para que essa percepção fosse possível, diversos estudos foram realizados, bem como entrevistas com os potenciais usuários da BO, as quais seguiram um instrumento de pesquisa. Ambos encontram-se

especificados no documento M1 - Diagnóstico da Base Organizacional [HPC06a]. O consenso nas opiniões coletadas, dos diferentes perfis de usuários entrevistados, deixou claro que o funcionamento e a utilização efetiva da BO dependem da modificação do atual processo de carga. Esse foi caracterizado como crítico devido a uma série de limitações, as quais estão descritas abaixo.

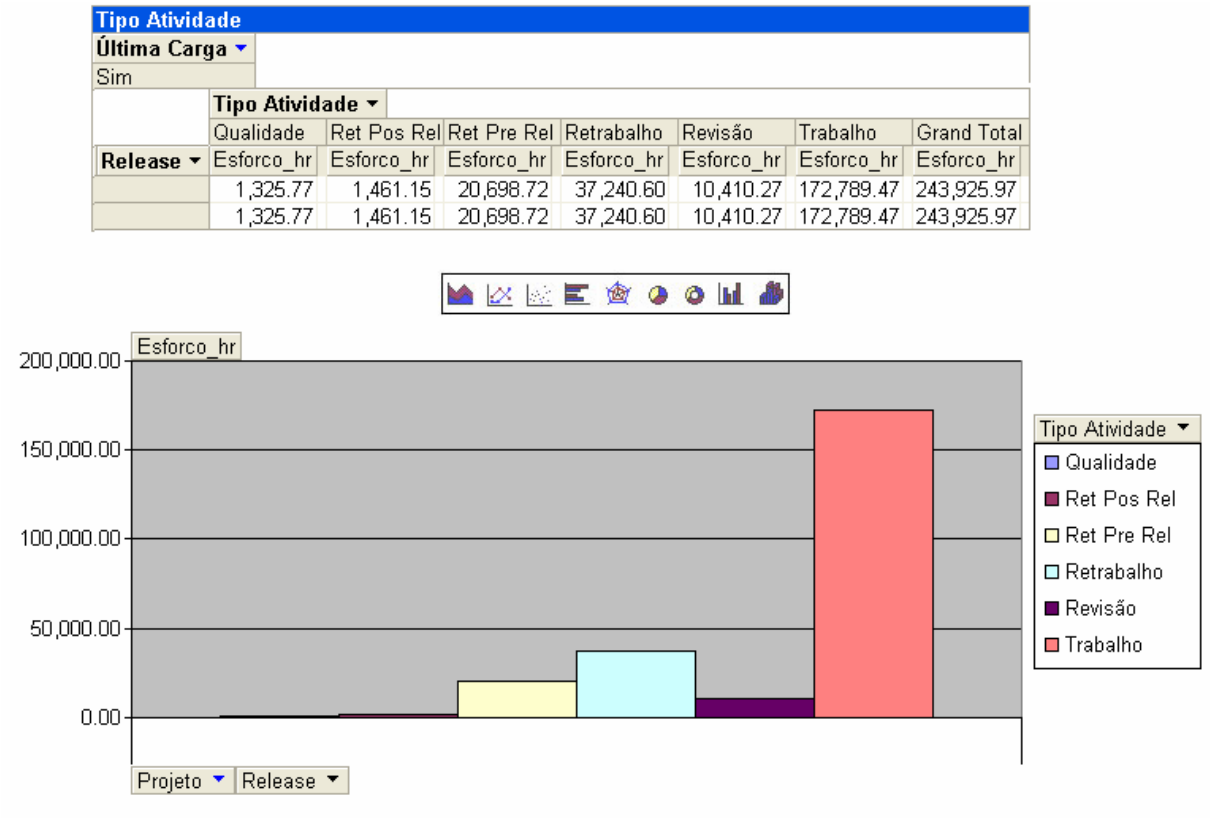


Figura 13 – Cubo e Recursos *OLAP*.

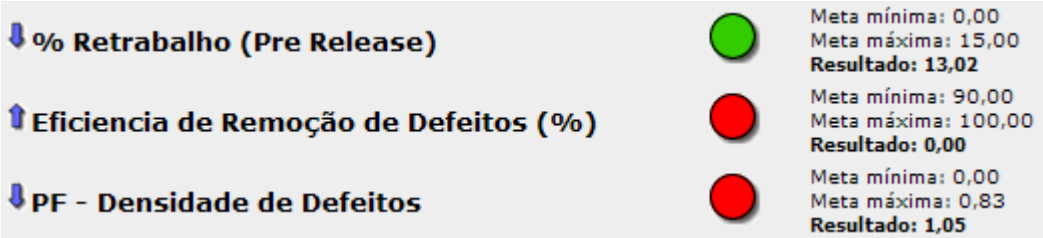


Figura 14 - Indicadores de Qualidade.

- *Processo de preparação dos dados laborioso e sujeito a erros:* O processo de obtenção dos dados necessários não é satisfatório, sendo caracterizado como laborioso e sujeito a erros. Isso se deve ao fato de ser altamente manual e, em alguns momentos, considerado como sobrecarga de atividades. Essa limitação foi apontada como crucial pelos usuários, tanto pelos

participantes do processo de preparação como pelos que têm conhecimento da forma pela qual o mesmo é desempenhado.

- *Confiabilidade dos dados implica em grande esforço e trabalho manual:* Devido ao fato da coleta das medidas ser realizada de forma manual, a confiabilidade dos dados coletados e apresentados pela interface de BI requer a execução de duas tarefas manuais de homologação, também consideradas sobrecarga de atividades e passível de inserção de erros.

- *Latência dos dados não é a desejada:* A dificuldade de preparação e consolidação dos dados faz com que essas informações sejam repassadas à BO mensal ou quinzenalmente. Devido a esse intervalo de tempo, os dados disponibilizados na interface de BI não refletem a situação atual do projeto, impedindo qualquer ação de monitoração.

- *Nível de Isolamento dificulta o processo:* A BO está localizada na Intranet da organização, enquanto que alguns projetos devem, por razões de segurança, trabalhar em sub-redes isoladas, sem acesso mesmo à Intranet. Sendo assim, para a coleta de qualquer métrica desse tipo de projeto, é inevitável alternar, fisicamente, as conexões com as redes ou usar algum tipo de mídia para transporte de arquivos. Isso dificulta o acesso e cria dificuldades adicionais no processo de preparação e envio dos dados.

- *Granularidade da informação disponibilizada nem sempre é a desejada:* A grande maioria dos entrevistados não está satisfeita com o nível de detalhamento das informações disponibilizadas. Muitos precisam de informações mais detalhadas e não conseguem ter acesso às mesmas. Alguns sugerem informações em granularidade maior (*e.g.* por projeto), enquanto outros, em granularidade menor (*e.g.* por atividade, e não por tipo de atividade (ver Seção 4.3.2)).

- *Não é oferecido suporte à Monitoração:* Entre os Gerentes de Projetos e Gerentes de *Software*, foi unânime a sugestão de se utilizar a BO para o monitoramento dos projetos em desenvolvimento. As atuais métricas baseadas em variações só produzem resultados significativos quando aplicadas sobre projetos concluídos, em situações de análise retrospectiva. Atualmente, esses usuários já realizam esforço para coletar as medidas, preparar e homologar os dados do processo de carga. Porém a BO não lhes oferece informações úteis para monitoração. Sendo assim, o benefício da BO não é claramente percebido, o que reforça a caracterização das atividades de preparação e inserção de dados na BO como sobrecarga. Para que a BO possa dar suporte à monitoração de projetos, os dados coletados, as métricas calculadas e os indicadores devem ser revistos.

- *Processo de Carga não é Incremental:* O atual processo de carga não é incremental. Sempre que um novo conjunto de dados precisa ser inserido ou alterado, todos os dados anteriores são recarregados. Sendo assim, o responsável pelo processo de carga deve manter um histórico contendo todos os arquivos de carga utilizados anteriormente. Essa prática oferece vários riscos à organização, além de ocorrerem inserções desnecessárias de registros já existentes dentro do banco de dados, ocupando espaço e dificultando o processo de extração de informação através de outras interfaces que não sejam a da Ferramenta de *BI*. Outro problema está relacionado às rotinas de *backup* para manter, com segurança, todos os arquivos utilizados em cargas anteriores.

5.3 Automatização do processo de ETC da Base Organizacional

Com base nas dificuldades encontradas no ambiente da BO constatou-se a necessidade de automatizar o processo de ETC das métricas, objetivando minimizar as tarefas manuais, executadas durante o processo de carga. Para tanto, foram realizados estudos para estimar o esforço necessário e os possíveis impactos causados pela automatização. As próximas seções descrevem brevemente: (i) o tempo total consumido pelo atual processo de carga; (ii) a solução computacional vigente desse processo; (ii) uma proposta de automatização para o mesmo; e (iii) as estimativas e o impacto dessa última.

5.3.1 Tempo Consumido pelo Processo de Carga

Para mensurar o tempo total consumido pelo processo de carga foi realizado o acompanhamento da sua execução no ambiente de um dos projetos da organização. O projeto escolhido utiliza como ferramentas de apoio à gestão: o *MS Project* (Cronograma), o *ClearQuest* (Qualidade), um *software* personalizado para coletar esforço realizado, algumas planilhas *MS Excel* e documentos *MS Word* (demais áreas de qualidade). A Tabela 7 apresenta os tempos despendidos durante a preparação dos dados, a serem armazenados em arquivos padronizados (Seção 5.1.1), de uma determinada versão (podem haver várias em aberto), com suas respectivas fontes de armazenamento. Já a Tabela 8 mostra os tempos despendidos durante a realização das atividades de homologação (conferência) e realização da carga (execução dos *DTS* de carga).

O somatório de tempos da primeira tabela é de 90 minutos por versão. Como, durante o acompanhamento da carga, existiam seis versões em andamento, o tempo total para a realização dessa etapa foi de 540 minutos. Assim, a partir das duas tabelas, constata-se que

foram despendidos 758 minutos para execução de todo o processo de carga, ou seja, 12h38min (três turnos de trabalho de um recurso). Ressalta-se que esses tempos são proporcionais ao número de versões em andamento, mas mesmo assim são considerados elevados e mal vistos pelos projetos. No entanto, salienta-se que esse processo de carga ainda é considerado conveniente à organização, já que o repositório de métricas fornece, somente, suporte à análise dos projetos, e que se admita uma maior latência dos dados. Nesse sentido, a organização entende que é economicamente aceitável deslocar recursos, tanto de projeto quanto do grupo de qualidade, das suas atividades normais para realizar o processo de coleta uma vez por mês.

Tabela 7 - Tempos de extração e preparação dos dados de carga.

Arquivos	Fontes de Armazenamento	Tempos (min)
Cronograma	*MS Project*	10
Esforço Realizado	Software personalizado	25
Defeitos	*ClearQuest*	25
Requisitos	MS Word	15
Informações Adicionais	*MS Excel e Word*	15

Tabela 8 - Tempos para a homologação e execução dos pacotes de carga.

Atividades	Ambiente/Ferramenta	Tempo (min)
Aceitar Pacote de Carga	*Site* BO	5
Conferir e Publicar Cronogramas	*MS Project*	133
Conferir Arquivo de Defeito	*MS Excel*	19
Conferir Arquivo de Esforço	*MS Excel*	9
Conferir Arquivo de Requisito	*MS Excel*	2
Mover Arquivos	*Windows Explorer*	12
Executar *DTS*	*SQL Server* 2000	30
Conferir Dados Interface de *BI*	*Site* BO	8

5.3.2 Processo de ETC Atual

O atual processo de ETC consiste na execução seqüencial de pacotes *DTS*, tendo como pré-condição que todos os dados brutos dos projetos sejam coletados, transformados e disponibilizados segundo o padrão organizacional especificado nos documentos de carga [HPC05a]. A Figura 15 representa o fluxo seqüencial de execução dos principais *DTS*, os quais contêm subfluxos onde são especificadas as ações necessárias para realizar: (i) a extração dos dados dos arquivos das diferentes áreas de qualidade; (ii) a transformação desses dados a partir de uma área de armazenando temporária (*DSA*, Seção 2.4); e (iii) a sua carga e a geração dos cubos, conforme a estrutura dimensional do repositório de métricas.

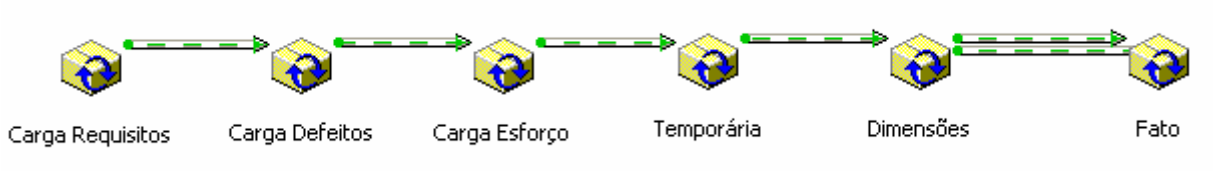


Figura 15 – Fluxo seqüencial de execução dos principais pacotes de *DTS*.

Cada um desses *DTS*s, apresentados na Figura 15, possui um escopo bem definido:

- Carga Requisitos: responsável por extrair, transformar e limpar as informações relacionadas com requisitos, disponibilizadas pelos projetos através de planilhas *MS Excel*, formatadas conforme o padrão organizacional especificado, e armazená-las no *DSA*.
- Carga Defeitos: realiza a extração, transformação, limpeza e cálculo das métricas referentes aos defeitos localizadas em planilhas *MS Excel*.
- Carga Esforço: trata os dados e as métricas relacionadas com o esforço realizado. No caso de um determinado projeto, esses dados de esforço são extraídos de planilhas *MS Excel*. Já em outro projeto, essas informações são retiradas diretamente da base de dados do *EPM*.
- Temporária: responsável por coletar os dados relacionados com os projetos (tamanho do time, custo, versão, tamanho, entre outras) e com o cronograma (atividades, *baselines*, fases, iterações, entre outros). Durante a execução desse pacote, esses dados são extraídos diretamente do *EPM* e colocados primeiramente em tabelas temporárias (*DSA*), onde passam por um processo de transformação e limpeza para depois serem armazenados no *DW*. Nesse pacote também é realizado o cálculo de algumas métricas derivadas da organização.
- Dimensões: Responsável por atualizar as dimensões com base nos dados carregados nas tabelas temporárias, alimentadas pelos *DTS*s anteriores.
- Fato: Responsável por construir as tabelas fato com base nas tabelas temporárias, alimentadas pelos *DTS*s anteriores.

5.3.3 Proposta de Automatização do Processo de ETC Atual

A proposta de automatização consiste na migração dos atuais *DTS* para *SQL Server Integration Services* (*SSIS*), utilizando *WS* para extrair os dados diretamente da fonte original,

localizada no ambiente dos projetos. Essa proposta é conseqüente do interesse da organização de continuar utilizando a interface de *BI*, a inteligência dos *DTS*, bem como a ferramenta de gestão de banco de dados oferecida pela *Microsoft*. Os SSIS acompanham a versão 2005 do *SQL Server* e seu uso é considerado a evolução para os *DTS* da versão 2000. Além disso, esses novos serviços possuem componentes especiais que facilitam a inclusão de *WS*, permitindo que os mesmos possam ser chamados durante a sua execução. Segundo a *Microsoft* [MSD07], os *DTS* estão sendo descontinuados e por essa razão não serão disponibilizados nas versões posteriores do *SQL Server*. Logo, os mesmos devem ser migrados.

Dentro desse contexto foram realizados estudos [HPC07a] [HPC07b] para migrar os *DTS* para *SSIS* e incorporar *WS* de extração. Os mesmos apresentam: (i) uma breve descrição da configuração do ambiente computacional necessário para editar, executar e testar os SSIS; (ii) os passos executados para realizar a migração dos *DTS* para *SSIS*; e (iii) a especificação dos *DTS* de Defeitos e Temporária e dos seus *SSIS* correspondentes, incluindo a funcionalidade de *WS*; e (iv) o esforço estimado e o impacto causado para realizar a migração completa do processo de ETC atual.

A Figura 16 tem por objetivo ilustrar os componentes envolvidos na construção de um *DTS* e destacar os pontos onde os *WS* devem ser inseridos. A mesma contém os componentes que constituem o *DTS* Temporária. As setas determinam a ordem do fluxo de execução dos mesmos. Os demais componentes do fluxo, representados por cilindros, são responsáveis por executar tarefas *SQL*, e os mini-diagramas caracterizam as atividades programadas em *scripts ActiveX*. As setas horizontais de cor preta, com rótulos Produto e Análise x Fase, realizam a importação e exportação de dados entre os componentes Base Project e Base Analítica, que representam respectivamente a base de dados do *EPM* e o repositório da BO. Portanto, os procedimentos dessas setas são os que devem ser substituídos por *WS*, para que a transferência dos dados possa ser realizada diretamente da fonte original para a área de armazenamento temporária (*DSA*).

A solução de automação proposta pelos estudos citados, baseada em *WS* e *SSIS*, exige que os *DTS* sejam migrados e adaptados. Para tanto, foi feito um estudo para avaliar o impacto dessa migração, envolvendo serviços de extração, dos pacotes de carga *DTS* Temporária e de Defeitos para o formato *SSIS*. Esses serviços foram codificados no *Visual Studio*, utilizando *J#* como linguagem de programação. Essa solução também contemplou a avaliação do impacto e do esforço associados com a evolução do Programa de Métricas [HPC06a] e do modelo analítico do repositório central [HPC06c]. Ambas são conseqüentes da

incorporação de métricas de monitoração. Cabe ressaltar que essas evoluções não são adotadas nesta pesquisa. As soluções citadas sugerem meramente o armazenamento das métricas derivadas, segundo os valores definidos pela *EVA*, e a inclusão de um novo fato, chamado Fato Demanda. O modelo analítico proposto pelo trabalho citado, mantém as mesmas limitações relacionadas com granularidade mencionadas na Seção 5.2, deste capítulo. Já esta pesquisa realiza apenas a extração das métricas diretas dessa técnica, calculando os seus valores durante o processo de sumarização dos cubos. Além disso, esta solução apresenta um modelo analítico mais elegante, com um número menor de tabelas fato, porém permitindo uma maior abrangência de perspectivas de análise e níveis de sumarização, o que a torna mais abrangente.

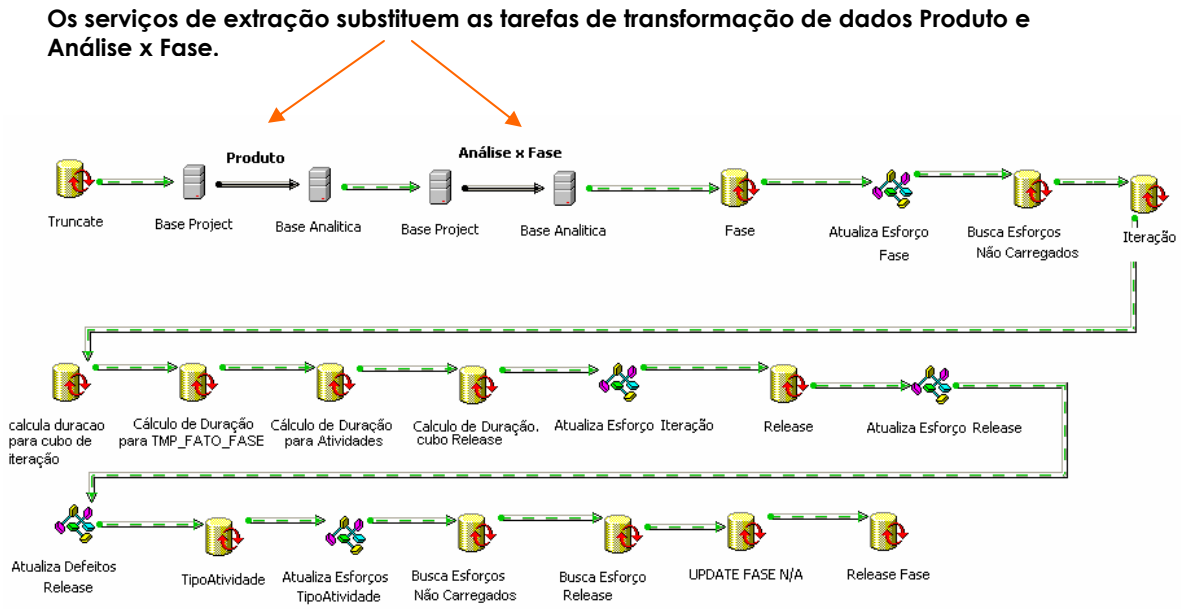


Figura 16 – *DTS* Temporária.

Como resultado do estudo citado [HPC07b] foi possível contabilizar o número de componentes que devem ser modificados, bem como classificar seus níveis de complexidade e dificuldade (*e.g.* Intermediário, Difícil). Essa classificação foi determinada com base na experiência de migração relatada em [HPC07a], para auxiliar a estimativa do esforço de migração de todos os pacotes de carga (Figura 15). A Tabela 9 mostra essa classificação, o número total de componentes que constituem o *SSIS*, a quantidade de componentes que precisa ser alterada ou incluída, e o esforço total de migração estimado em horas. Assim, contabilizando os esforços apresentados nessa tabela obtêm-se uma estimativa de aproximadamente 900 horas de desenvolvimento.

Tabela 9 - Estimativas de Impacto e Esforço [HPC07a].

Nome (*SSIS*)	Classificação	Total de Componentes	Componentes Alterados	Componentes Incluídos	Esforço (horas)
Carga Requisitos	Intermediário	11	2	3	120
Carga Defeitos	Difícil	12	2	7	180
Carga Esforço	Difícil	12	2	7	180
Temporária	Difícil	23	2	9	180
Dimensões	Intermediário	22	2	4	120
Fato	Intermediário	12	2	1	120

5.4 Considerações sobre o Estudo de Caso

Com base no que foi apresentado neste capítulo, percebe-se que o atual ambiente da BO não satisfaz a empresa parceira, principalmente no que se refere ao processo, laborioso e manual, de captura das métricas e a ausência do suporte à monitoração. Ambos são, respectivamente, conseqüentes de um processo semi-automatizado de captura das métricas e da ausência de métricas adequadas para monitoração, bem como de um modelo analítico com uma granularidade maior que a desejada. Além disso, cabe salientar que essa operação de *software* pretende atingir os níveis mais altos do *CMMI*. Por essa razão, é essencial que essas dificuldades sejam minimizadas, para que o repositório central possa oferecer suporte à monitoração, e conseqüentemente a uma gestão quantitativa, conforme o Nível 4. Para tanto, é primordial realizar cargas freqüentes e com um baixo nível de intrusão.

A proposta de migração dos *DTS* para *SSIS*, incluindo *WS* e incorporando serviços de extração, mostra-se adequada para minimizar o esforço de carga e permitir a monitoração. No entanto, essa solução não trata todos os aspectos de heterogeneidade, freqüência e latência mencionados na Seção 3.1. Além disso, a adoção dessa proposta foi suspensa, devido a recente aquisição, pela *HP Company*, da *Mercury Interactive*, que licencia um conjunto de ferramentas de apoio à gestão do desenvolvimento de *software*. Por essa razão, a gerência da operação brasileira ainda não se pronunciou sobre qual encaminhamento vai ser dada à BO como instrumento de gestão de projetos.

Portanto, conclui-se que as características e as dificuldades da operação de *software*, apresentadas neste capítulo, serviram para corroborar as dificuldades levantadas no *SPDW*, reportadas na Seção 4.3, e enfatizar a necessidade de procurar uma solução que enderece os aspectos computacionais mencionados no Capítulo 2.

6 SOLUÇÃO

Este capítulo descreve a solução computacional para automatizar a mensuração da qualidade de *software* denominada *SPDW+*. É uma evolução substancial do *SPDW*, proposto por [BEC06] e [CUN05] por, principalmente, dar suporte à monitoração. Nesse sentido, as próximas seções contemplam: (i) o método de pesquisa utilizado; (ii) a definição do programa de métricas suportado pela solução; (iii) a determinação do modelo analítico; (iv) uma visão geral da arquitetura do *SPDW+*; (v) a especificação do processo automatizado de ETC das métricas, juntamente com as camadas que compõem a arquitetura e as soluções adotadas para os seus componentes principais; e (vi) a verificação da qualidade da solução, segundo o instrumento de referência (Seção 3.2).

6.1 Método de Pesquisa

Segundo [FAC02] “todo trabalho científico deve ser apreciado em procedimentos metodológicos”. Quanto aos objetivos, classifica-se esta pesquisa como exploratória, pois foi realizada a identificação de um problema e tem-se como finalidade a descoberta de teorias e práticas a serem aplicadas no ambiente real, pretendendo modificar as ações existentes e buscando oferecer as inovações tecnológicas necessárias para solucionar as dificuldades identificadas. Quanto ao procedimento, inicialmente foi realizado um estudo de um ambiente real em uma operação de *software* certificada *CMM3*, onde foram constatadas limitações importantes no monitoramento dos seus projetos em andamento, devido à dificuldade de obter métricas atualizadas, de maneira automatizada, considerando o ambiente heterogêneo e dinâmico do PDS. Ainda, foi construído um ambiente controlado, em laboratório (laboratório GPIN), com a intenção de simular o ambiente real, onde foram realizados testes para avaliar a qualidade da solução proposta.

6.2 Programa de Métricas

Para oferecer um suporte às atividades de supervisão da qualidade de *software* (Seção 3.1.2) é necessário estabelecer um conjunto de métricas adequado, a partir do qual seja possível realizar análises, monitorações e previsões com base nessas métricas de *software*. A Tabela 10 apresenta o Programa de Métricas estendido, originalmente proposto por [BEC06], destacando (em sombreado) as métricas de monitoração incorporadas, definidas segundo a

técnica de *EVA* (Seção 2.3). Os itens subseqüentes desta seção relatam, respectivamente, a inclusão de *EVA* no Programa de Métricas e uma demonstração de monitoração de prazo e custo, conforme essa técnica.

6.2.1 Programa de Métricas com *EVA*

Segundo a técnica de *EVA*, só se pode monitorar ao longo do tempo algo que apresente uma linha base, e que permita uma comparação entre o inicialmente planejado e o efetivamente realizado, verificando se o seu desempenho apresenta variações dentro de uma faixa de valores considerada aceitável. Assim, com base nessas exigências foi feita uma análise das áreas de qualidade originais (Seção 4.3.1), buscando detectar quais poderiam ser monitoradas.

Com base nos resultados dessa análise, concluiu-se que:

- *Tempo*: as métricas desta área são fundamentais à análise do valor agregado, pois são responsáveis por determinar o intervalo cronológico em que determinada tarefa deve ser realizada. Por esse motivo, as métricas de tempo estão diretamente relacionadas com as demais áreas que podem ser monitoradas.

- *Esforço* e *Custo*: as métricas da área de esforço apresentam as características desejáveis à realização da análise do valor agregado: esforço planejado e real, dentro de um intervalo de tempo definido por um cronograma. Logo, é conveniente verificar se o esforço realizado está conseguindo agregar o valor desejado à determinada tarefa, produto ou projeto. Assim como as de esforço, as métricas de custo detêm as características desejáveis para que se possa aplicar a *EVA*.

- *Tamanho*: esta área contém métricas de tamanho estimado e real, dentro de um intervalo de tempo pré-estabelecido. Contudo, durante o desenvolvimento do código de um componente de *software*, é difícil ter-se a informação do percentual de tamanho completado, pois a mesma depende muito da técnica de escrita de código que o desenvolvedor usa bem como do tipo de ambiente de programação adotado. Logo, a mesma não apresenta os pré-requisitos necessários suportar a monitoração.

- *Requisitos*: as métricas desta área controlam a variação do número de requisitos inicialmente acordados com os clientes. Durante a construção de um determinado

produto, o cliente pode adicionar ou excluir funcionalidades conforme as suas necessidades e prioridades. Esse comportamento interfere no desempenho da execução do cronograma, pois o planejamento inicial precisa ser refeito e, em alguns casos, o trabalho realizado acaba sendo computado, mesmo sem agregar valor ao produto, podendo gerar atraso na data de entrega. No entanto, tipicamente não é realizado um controle ou avaliação do percentual de completude de um requisito, fundamental à determinação do valor agregado.

Tabela 10 – Programa de Métricas do *SPDW+*.

Áreas de Qualidade	Métricas Derivadas	Métricas Diretas
Tempo	**VBO –** Variação do *Baseline* Original **VBR –** Variação do *Baseline* Revisado	**DIR –** Data Inicial Real **DFR –** Data Final Real **DIBO** – Data Inicial do *Baseline* Original **DFBO –** Data Final do *Baseline* Original **DIBR –** Data Inicial do *Baseline* Revisado **DFBR –** Data Final do *Baseline* Revisado
Esforço	**VEBO –** Variação de Esforço do *Baseline* Original **VEBR –** Variação de Esforço do *Baseline* Revisado **PR –** Produtividade	**ER –** Esforço Real **EBO –** Esforço do *Baseline* Original **EBR –** Esforço do *Baseline* Revisado **TR –** Tamanho Real
Tamanho	**VTBO –** Variação de Tamanho do *Baseline* Original **VTBR –** Variação de Tamanho do *Baseline* Revisado	**TBO –** Tamanho do *Baseline* Original **TBR –** Tamanho do *Baseline* Revisado **TR –** Tamanho Real
Custo	**VC –** Variação de Custo **VCA –** Variação de Custo Agregada **VPA** – Variação de Prazo Agregada **IDC –** Índice de Desempenho de Custo **IDP –** Índice de Desempenho de Prazo	**CR –** Custo Real **CBO –** Custo do *Baseline* Original **CBR –** Custo do *Baseline* Revisado **CRAR –** Custo Real da Atividade de Revisão **CRFT –** Custo Real da Fase de Teste **CRAQ –** Custo Real das Atividades de Qualidade **CRART –** Custo Real das Atividades de Retrabalho **%TC –** %Trabalho Completado **DS –** Data de *Status*
Requisitos	**VR –** Volatilidade de Requisitos	**NMA** – Nro. de Modificações Aprovadas **NMR** – Nro. de Modificações Requeridas **NRE –** Nro. de Requisitos Excluídos **NRM –** Nro. de Requisitos Modificados
Qualidade	**ERD –** Eficiência de Remoção de Defeitos **DDE –** Densidade de Defeitos Entregues **DDI –** Densidade de Defeitos Internos **ERV –** Eficiência de Revisão **SC –** Satisfação do Cliente	**NDI –** Nro. de Defeitos Internos **NDE –** Nro. de Defeitos Externos **TR –** Tamanho Real **ISC –** Índice de Satisfação do Cliente

- <u>*Qualidade*</u>: as métricas desta área não apresentam valores estimados durante um espaço de tempo pré-definido. Além disso, a métrica **NDE** (Número de Defeitos

Externos) considera valores detectados quando o produto encontra-se em funcionamento no ambiente do cliente. Logo, não existe um intervalo de tempo definido para a captura dessas informações. Por essa razão, o método de análise do valor agregado não pode ser aplicado.

A partir dessas conclusões optou-se por incluir métricas para monitorar *Custo* e *Tempo*. Considerou-se *Esforço* um caso particular da primeira (quando os recursos alocados para realizar uma determinada tarefa apresentam custos equivalentes) e, ao considerar *Custo*, entende-se que *Esforço* também está sendo coberto. A área de qualidade *Tamanho* não foi incluída devido à dificuldade em coletar e tratar adequadamente o percentual de tamanho completado. Já as áreas de qualidade *Requisitos* e *Qualidade* (Defeitos e Satisfação do Cliente) também não foram incluídas porque dificilmente apresentam valores estimados e parciais.

Logo, para oferecer suporte à monitoração, de prazo (Tempo) e custo, utilizando a técnica de *EVA* foram definidas as seguintes métricas:

- Diretas:
  - Percentual de Trabalho Completado (**%TC**)
  - Data de *Status* (**DS**)
- Derivadas:
  - Índice de Desempenho de Prazo (**IDP)**
  - Índice de Desempenho de Custo (**IDC)**
  - Variação de Prazo Agregada (**VPA**)
  - Variação de Custo Agregada (**VCA**)

A Tabela 11 apresenta a definição dessas métricas derivadas, juntamente com o seu objetivo, as métricas diretas utilizadas para o seu cálculo, bem como a sua equação e a sua unidade de tamanho.

Tabela 11 – Métricas derivadas para monitorar custo e prazo.

Métricas Derivadas	Objetivos	Métricas Diretas	Equação	Unidade
Variação de Prazo Agregada (**VPA**)	Fornecer a relação entre o *baseline* (original ou revisado), o *baseline* real e o percentual de trabalho completado até a data de *status*.	Data Final *Baseline* Original (**DFBO**) Data Inicial *Baseline* Original (**DIBO**) Data de *Status* (**DS**) Custo Baseline Original (**CB**) % Trabalho Completado (**TC**)	$VPA = CBO \times \left(\frac{TC}{100} - \frac{(DS - DIBO)}{(DFBO - DIBO)} \right)$	%
Variação de Custo Agregada (**VCA**)	Fornecer a relação entre o custo (original ou revisado), o custo real e o percentual de trabalho completado até a data de *status*.	Custo *Baseline* Original (**CBO**) % Trabalho Completado (**TC**) Custo Real (**CR**)	$VCA = \frac{CBO \times TC}{100} - CR$	%
Índice de Desempenho de Prazo (**IDP**)	Fornecer a taxa de conversão do valor estimado em valor agregado.	% Trabalho Completado (**TC**) Data Final *Baseline* Original (**DFBO**) Data Inicial *Baseline* Original (**DIBO**) Data de *Status* (**DS**)	$IDP = \frac{TC \times (DFBO - DIBO)}{100 \times (DS - DIBO)}$	%
Índice de Desempenho de Custo (**IDC**)	Fornecer a relação entre os custos consumidos pelo projeto e valor agregado.	Custo *Baseline* Original (**CBO**) % Trabalho Completado (**TC**) Custo Real (**CR**)	$IDC = \frac{CBO \times TC}{100 \times CR}$	%

6.2.2 Monitoração de Custo e Prazo segundo *EVA*

Esta seção apresenta um exemplo da monitoração de custo e prazo, segundo a técnica de *EVA*. Inicialmente, é apresentado o cronograma de um projeto fictício, juntamente com os dados estimados e as métricas de monitoração (Tabela 12). Por fim, é relatada a interpretação desses valores.

A Tabela 12 ilustra o planejamento inicial desse projeto fictício, com duração de um ano e contendo uma única versão, composta por diversas fases, formadas por um conjunto de tarefas. As colunas dessa tabela apresentam os valores estimados (**VE** e **VE DS**), o valor agregado (**VA**), as métricas diretas (**%TC** e **CR**) e derivadas (**VPA**, **VCA**, **IDP** e **IDC**). Para cada uma das Tarefas é realizado um planejamento inicial de custo (**VE**). Com relação ao prazo, supondo-se uma distribuição linear do orçamento, espera-se que no término do primeiro trimestre as atividades estejam 25% concluídas; no segundo, 50%; no terceiro, 75%; e, no quarto, 100%. Cabe salientar que essa tabela apresenta valores correspondentes ao terceiro mês de execução do projeto (data *status*), e respectivos valores estimados (**VE DS**).

Tabela 12 – Exemplo de cronograma monitorado com *EVA*.

Nível	VE	VE DS	%TC	VA	CR	VPA	VCA	IDP	IDC
Projeto	2660,00	665,00	28,8	765,80	891,00	100,80	-125,20	1,2	0,9
Versão	2660,00	665,00	28,8	765,80	891,00	100,80	-125,20	1,2	0,9
Fase	2660,00	665,00	28,8	765,80	891,00	100,80	-125,20	1,2	0,9
Tarefa 1	896,00	224,00	25,0	224,00	400,00	0,00	-176,00	1	0,6
Tarefa 2	672,00	168,00	50,0	336,00	80,00	168,00	256,00	2	4,2
Tarefa 3	448,00	112,00	10,0	44,80	250,00	-67,20	-205,20	0,4	0,2
Tarefa 4	644,00	161,00	25,0	161,00	161,00	0,00	0	1	1

As células em amarelo, da Tabela 12, correspondem às tarefas rigorosamente no prazo e no custo inicialmente planejado. Por sua vez, as em verde significam que os níveis encontram-se adiantados ou abaixo do orçamento previsto. Já as em vermelho representam níveis em atraso ou acima do custo estimado.

Com base nos valores referentes à Tarefa 1 (Tabela 12) é possível construir o gráfico da Figura 17. Por intermédio do mesmo, percebe-se que essa tarefa está sendo realizada conforme o prazo planejado, pois as curvas **VE** e **VA** se sobrepõem no ponto determinado pela data de *status*. Logo, pode-se concluir que essa tarefa foi executada exatamente no prazo planejado (25% concluída). Porém a mesma está sendo efetuada com um orçamento maior do que o estimado. Isso pode ser constatado através da análise das curvas **CR** e **VE**, onde a primeira apresenta uma distância positiva em relação à segunda, tendo como referencial o eixo das ordenadas. Por essas razões, a variação de prazo (**VPA**) apresenta valor zero, enquanto que a variação de custo (**VCA**) é negativa. Maiores detalhes sobre a obtenção dos valores, que compõem a Tabela 12, e a interpretação dos mesmos podem ser encontrados na Seção 2.3.

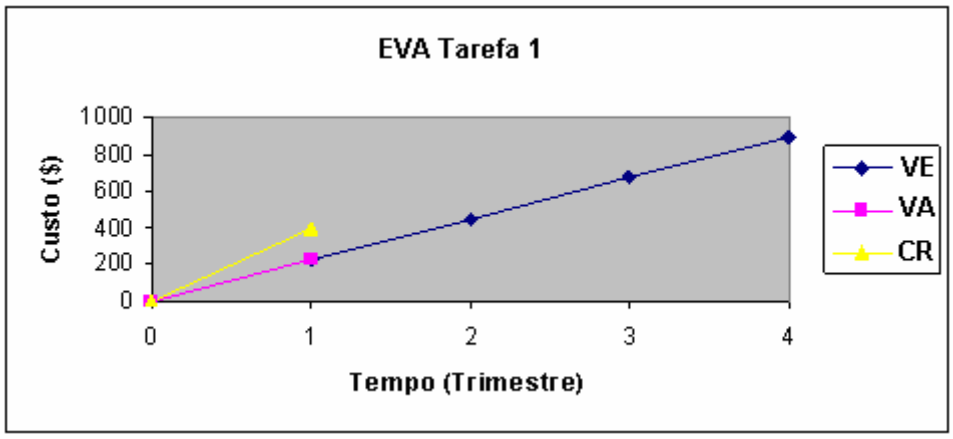


Figura 17 – Gráfico *EVA* da Tarefa 1.

6.3 Modelo Analítico

O modelo analítico descrito nesta seção é conseqüente do Programa de Métricas e dos modelos de processo de desenvolvimento de *software*, representados pela estrutura de projetos, relatada na Seção 6.3.1. É relatado em detalhes o modelo dimensional proposto, e é discutida a necessidade de adoção do período de validade para os fatos, baseada em técnicas de banco de dados temporais, e mostrada a solução adotada.

6.3.1 Estrutura de Projetos

A Figura 18 mostra a estrutura de projetos adotada, representada em um diagrama de classe UML. A definição dessa estrutura considera o conteúdo do Programa de Métricas (Seção 6.2) e os pontos dos modelos de processo de desenvolvimento de *software* onde as métricas devem ser coletadas. Tal estrutura permite representar modelos de processos com ciclo de vida cascata e iterativo, considerados clássicos e amplamente utilizados [PRE04]. Além disso, a mesma também admite variações desses ciclos que contenham pequenas diferenças relacionadas com: o nome, a ordem e a relação entre as fases ou os artefatos adotados. Por exemplo, cascata com sobreposição, espiral e evolutiva, citadas na Seção 2.1.1. Dessa forma, esta solução busca tratar de maneira adequada às particularidades relacionadas com a heterogeneidade dos modelos de processo de *software*, relatadas na Seção 3.1.3.2.

A estrutura adotada permite que cada projeto contenha uma ou mais versões, sendo que as mesmas podem apresentar um conjunto de iterações ou fases. Porém, uma fase pode ou não pertencer a uma iteração. Essa condição somente é possível devido à presença da classe estágio. As fases contêm atividades classificadas segundo seu tipo: trabalho, retrabalho,

revisão e qualidade. Os defeitos são mensurados a partir das fases e devem ser classificados conforme o seu grau de severidade (*e.g.* alto, médio ou baixo). Os atributos das classes do diagrama correspondem às métricas diretas e derivadas da Tabela 10, representando os níveis de detalhe em que as mesmas podem ser capturadas, ou produzidas. Essa estrutura também admite mais de uma hierarquia de cronograma: orientado a fases e orientado a entregáveis. Maiores detalhes sobre essas hierarquias e os ciclos de vida, com seus respectivos impactos, podem ser encontrados em [CUN05].

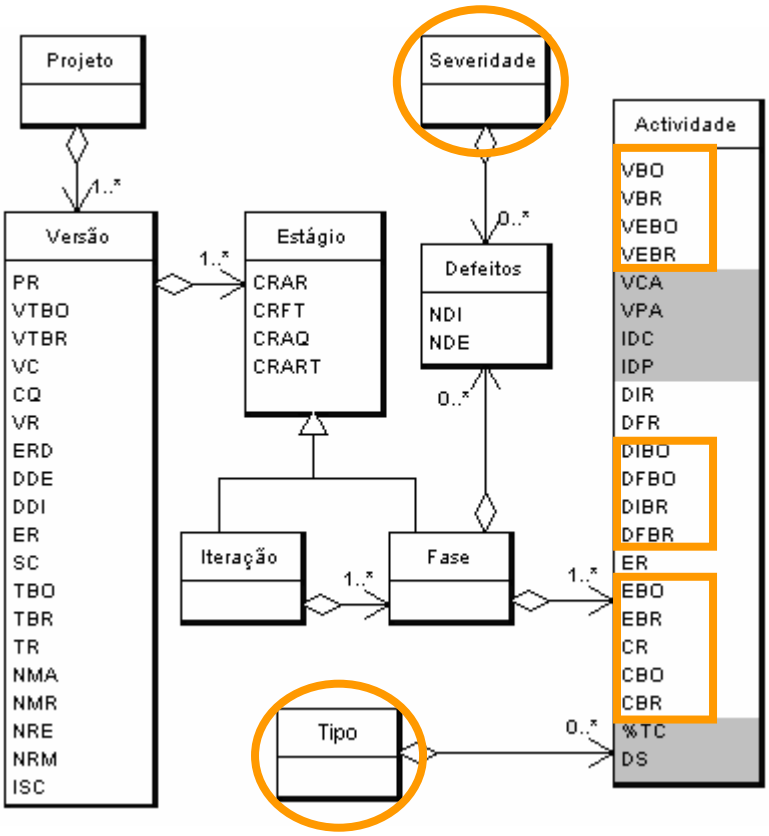


Figura 18 - Estrutura de Projetos e as Métricas do *SPDW+*.

Os aspectos ressaltados nessa estrutura (Figura 18) representam: (i) as métricas para oferecer suporte à monitoração, destacadas em sombreado; (ii) as métricas coletadas no grão de atividade, delimitadas por retângulos; e (iii) as classes Severidade e Tipo, enfatizadas por círculos. O primeiro aspecto permite que os projetos e a organização tenham seus custos e prazos monitorados a partir do nível de atividade. Já o segundo possibilita uma supervisão mais detalhada dos projetos e oferece subsídios para que a primeira seja realizada. Por sua vez, o terceiro admite, por um lado, classificar os defeitos conforme ao seu grau de severidade, permitindo que novas categorias possam ser adicionadas quando necessário, e, por

outro lado, torna possível, pela classe Tipo, a classificação de novas atividades, permitindo que outras categorias sejam criadas, sem que isso comprometa a estrutura de projeto. Uma situação real ilustrativa foi a nova classificação de retrabalho, adotada pela operação de *software* relatada no Capítulo 5, em que foi necessário definir novos valores para as atividades de retrabalho: Pré-Release (Ret-Pré-Rel) (realizado durante o desenvolvimento do produto) e Pós-Release (Ret-Pós-Rel) (efetuado quando o produto encontra-se em produção). É importante mencionar que os aspectos ressaltados contribuem significativamente para tornar a solução mais flexível e abrangente, pois possibilitam que novas categorias possam ser inseridas e a monitoração de custo e prazo possa ser feita a partir do nível de atividade.

6.3.2 Modelo Dimensional

O modelo analítico (Figura 19), substancialmente aprimorado em relação ao proposto pelo *SPDW* (Figura 11), mantém a estrutura dimensional do tipo constelação de fatos e oferece suporte à análise e monitoração. Para ser legível, o modelo ilustrado nesta seção contém, apenas, as áreas de qualidade que correspondem às métricas armazenadas nas tabelas fato, brevemente, descritas na Tabela 13. Já as tabelas dimensão são representadas apenas pelo seu nome e encontram-se descritas na Tabela 14. Maiores detalhes sobre o embasamento para a definição desse tipo de modelo de dados dimensional podem ser encontrados na Seção 2.4.1.

Com relação à estrutura das tabelas fato, esta proposta apresenta somente métricas aditivas (*e.g.* esforço (**ER**) e custo (**CBO**)) e semi-aditivas (*e.g.* percentual de trabalho completado (**%TC**)) nos seus atributos, fundamentais para possibilitar sumarizações entre as diferentes dimensões [KIM98]. Dessa forma, com somente três tabelas fato (Fato_Atividade, Fato_Versão e Fato_Defeito), tem-se diferentes perspectivas de análise e níveis de sumarização (iteração, fase, tipo do fato), de uma maneira elegante. Os fatos também são compostos por chaves estrangeiras, responsáveis por manter a integridade referencial das dimensões e possibilitar diferentes perspectivas de análise e níveis de sumarização. É importante salientar que as métricas derivadas não fazem parte da estrutura das tabelas fato e são calculadas no momento da criação dos cubos (Seção 7.4). Maiores detalhes sobre a estrutura das tabelas fato e dimensão e dos seus relacionamentos podem ser encontrados no Apêndice A.

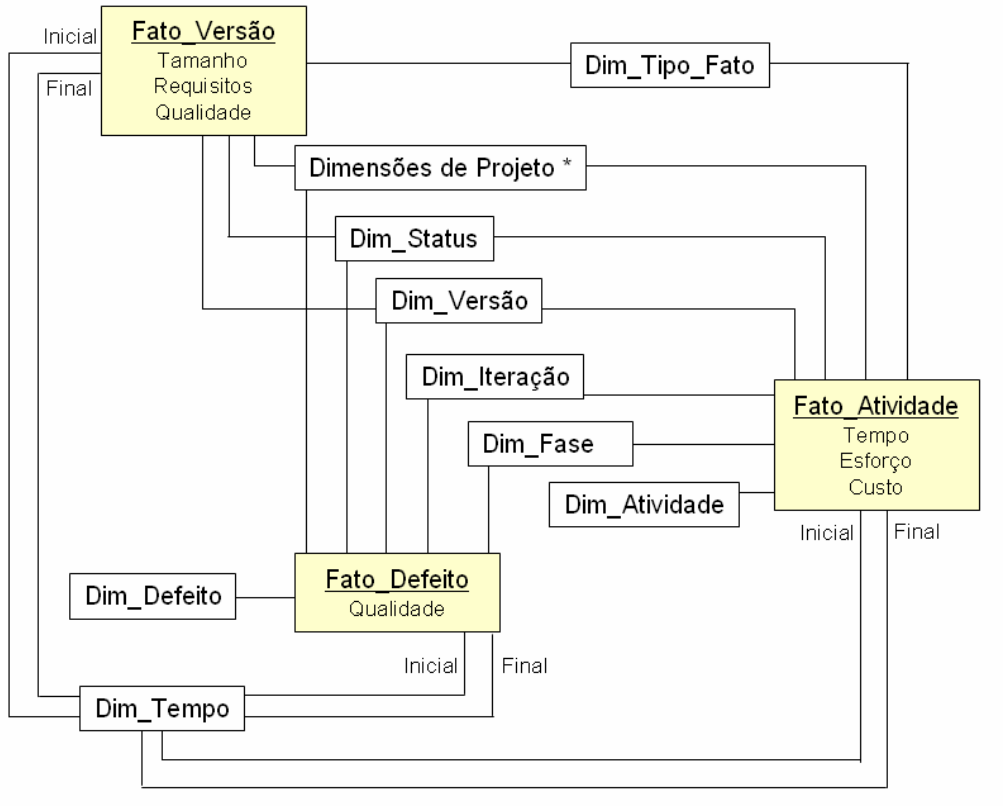


Figura 19 - Modelo Analítico do *SPDW+*.

Tabela 13 – Fatos do Modelo Analítico Estendido.

Tabela Fato	Descrição
Fato_Versão	Armazena métricas (originais, revisadas e reais) de Requisitos, Qualidade e Tamanho de uma versão de projeto.
Fato_Defeito	Armazena métricas de Defeitos (internos e externos) de uma versão em determinada fase.
Fato_Atividade	Armazena métricas (originais, revisadas e reais) de Esforço, Cronograma e Custo das atividades.

Tabela 14 – Dimensões do Modelo Analítico Estendido.

Dimensão	Descrição
Dimensões de Projeto	Corresponde a distintas dimensões com propriedades de projetos, que por razões de legibilidade, foram agrupadas no desenho. As dimensões são: Projeto, Indústria, Cliente, Porte, Tipo (manutenção ou desenvolvimento) e Tecnologia.
Dim_Versão	Dados que caracterizam as versões do projeto.
Dim_Iteração	Dados que caracterizam iterações de versões de projeto.
Dim_Fase	Dados que caracterizam fases de versões de projeto.
Dim_Atividade	Dados que caracterizam atividades das fases do projeto, classificadas de acordo com seu tipo (trabalho, retrabalho (Ret-Pré-Rel e Ret-Pós-Rel), revisão, qualidade).
Dim_Defeito	Dados que caracterizam defeitos de fases de versão, baseados em categoria (interno, externo) e severidade (baixa, média e alta).
Dim_Status	Dados que caracterizam status do projeto, versão, iteração ou fase (em andamento ou finalizado).
Dim_Tempo	Dados que caracterizam a data inicial e final do fato.
Dim_Tipo_Fato	Define se o fato é uma estimativa (*baseline* original, *baseline* revisado) ou registro de uma realização (real).

Com relação às tabelas dimensão, vale ressaltar Dim_Tipo_Fato, composta por uma chave primária e um atributo categórico, contendo o tipo do fato: original (planejado), revisado (replanejado) e real (efetivamente realizado). Assim, a partir dessa dimensão, é possível oferecer essas três perspectivas de análise e, além disso, sumarizá-las em diferentes níveis, de maneira intuitiva e transparente.

A realização da monitoração requer um acompanhamento regular dos custos e prazos associados para cada atividade, conforme as particularidades exigidas pela supervisão do ambiente de desenvolvimento de *software*, descritas na Seção 3.1.2. Para tanto, o modelo proposto inclui métricas das áreas de qualidade: Esforço, Tempo e Custo, no nível de atividade.

6.3.3 Validade dos Fatos

Para oferecer um suporte adequado à monitoração da mensuração da qualidade de *software*, é aceitável que a coleta das métricas apresente uma freqüência e uma latência na ordem de horas (Seção 3.1.3.1). Além disso, ainda é necessário disponibilizá-las de maneira consolidada, consistente e atualizada a partir de um repositório central de métricas. Por outro lado, esse repositório também deve manter um histórico dessas métricas. Para tanto, [KIM98] sugere que apenas sejam mantidos dados novos ou atualizados no DW. Nesse sentido, para tratar essa última exigência, esta pesquisa propõe a adoção de período de validade para os fatos, baseado no proposto em [NAV93].

Conforme a definição de banco de dados temporais e do modelo relacional temporal, para estabelecer validade aos dados é obrigatório definir dois atributos temporais (*timestamps*): data de validade inicial (D_I) e data de validade final (D_F). Ambos correspondem, respectivamente, aos limites inferior e superior de um intervalo: um fato é considerado válido em todo o período compreendido entre D_I e D_F. Além disso, cada fato pode ter valor para D_I, porém valor desconhecido para D_F, no instante da sua criação. O uso de um valor infinito para representar essa situação pode resultar na errônea interpretação de que o fato é válido em qualquer instante de tempo no futuro. Por essa razão, é sugerido que quando um evento ocorre no presente e, o seu término não é conhecido, sua D_F não tenha valor válido (seja *null*).

Com base nessas definições e conceitos, esta solução determina que a D_I receba o valor da data da coleta e D_F seja nulo, até que o dado deixe de ser válido no presente e receba uma data de validade final. Dessa forma, o repositório não armazena dados repetidos,

mantendo apenas os fatos novos ou atualizados, permitindo a realização da carga de maneira incremental, de acordo com as regras criadas para controlar inclusões, atualizações e término da validade dos fatos, detalhadas na Seção 6.5.4. Tanto a estrutura das tabelas fato, quanto os atributos de validade, estão no Apêndice A.

6.3.4 Considerações sobre o Modelo Analítico

O modelo analítico proposto nesta seção amplia significativamente as vantagens já mencionadas para o *SPDW* (Seção 4.3), pois oferece: (i) suporte à monitoração; (ii) perspectivas de análises a partir do nível de atividade, para as áreas de qualidade: Tempo, Esforço e Custo; (iii) um número reduzido de tabelas fato; (iv) sumarizações nos diversos níveis, a partir de métricas aditivas e semi-aditivas; e (v) tratamento temporal para a validade dos fatos.

6.4 *SPDW+*

O *SPDW+* é um ambiente de *data warehousing* que oferece suporte para análise e monitoração da mensuração da qualidade de *software*, a partir de um Repositório de Dados, estruturado conforme o modelo analítico da Seção 6.3, e de um processo automatizado de ETC das métricas, orientado a serviço. Para tanto, esta solução trata os seguintes aspectos de automação computacional, definidos na Seção 3.1.3: (i) latência dos dados; (ii) freqüência de realização do processo de ETC das métricas de *software*; (iii) heterogeneidades de ferramentas, modelos de processo e tipos de projetos; e (iv) baixo nível de intrusão.

A arquitetura do *SPDW+*, ilustrada pela Figura 20, está organizada em camadas: Integração de Aplicações, Integração dos Dados e de Apresentação. A próxima seção apresenta o processo automatizado de ETC das métricas, realizado ao longo dessas camadas, especificando a solução adotada para resolver os seus principais componentes. Cabe salientar que a Camada de Apresentação não é detalhada, visto que os seus principais componentes não fazem parte deste tema de pesquisa.

6.5 Processo Automatizado de ETC

A solução adotada para definir o processo automatização de ETC de métricas, orientado a serviços, baseia-se em *WS* e nos conceitos de AOS, apresentados na Seção 2.5, pois ambos são considerados a tendência de mercado para atender às necessidades

relacionadas à flexibilidade e adaptabilidade de processos de negócio [EMI06] [KRO05] [CAS02a]. Esta seção descreve a solução, a partir das camadas da arquitetura do *SPDW+*, destacando os seus principais componentes: (i) a Camada de Integração de Aplicações, juntamente com a definição dos *Wrappers*, Metadados de Projeto e Rotina de Extração; (ii) a Camada de Integração dos Dados, contemplando a *DSA e* as Rotinas de Limpeza e Transformação; (iii) a carga incremental e as regras utilizadas; e (iv) o Repositório de Dados, implementado na forma de um *DW* e visualizado por intermédio de um cubo *OLAP*.

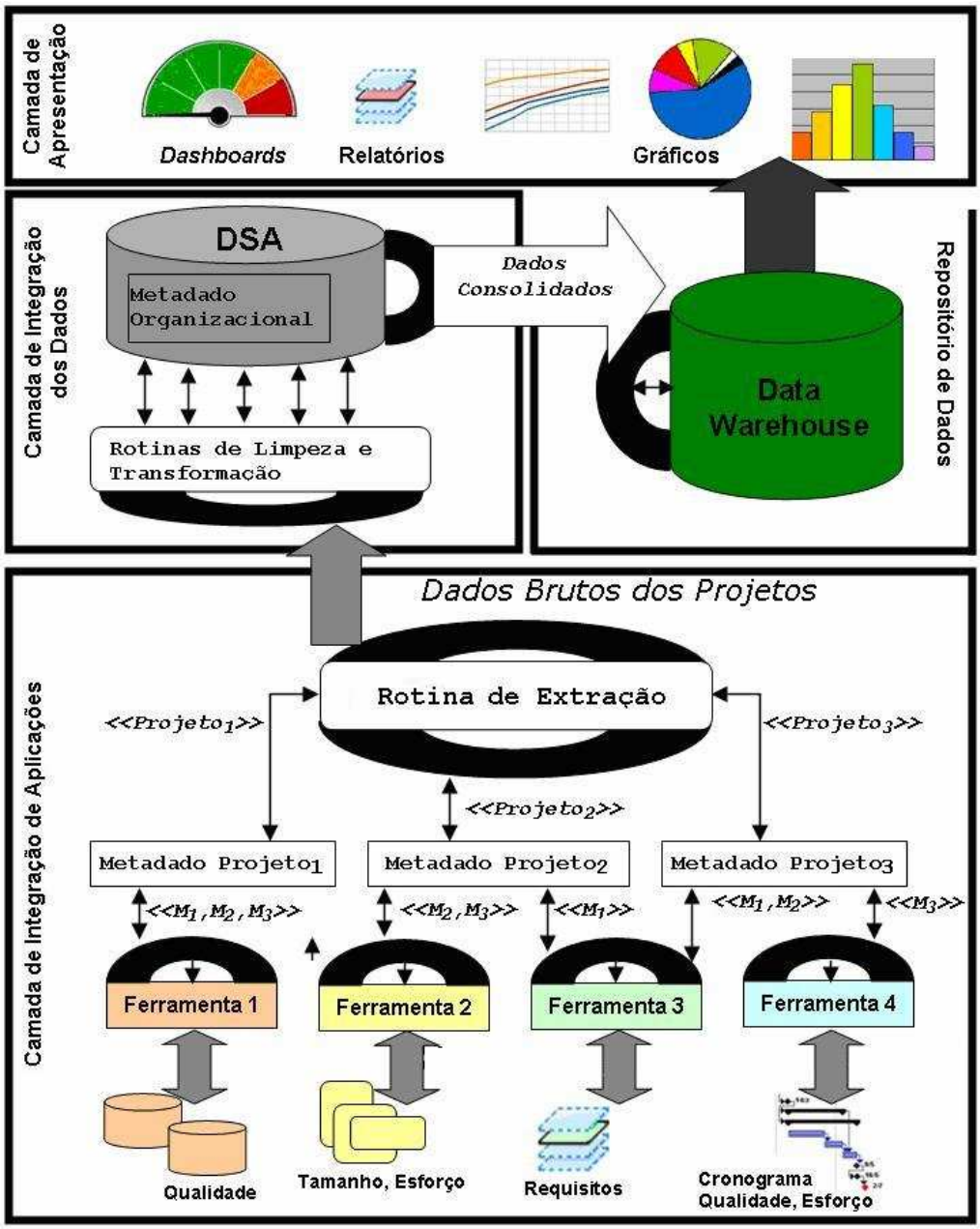


Figura 20 - Arquitetura do *SPDW+*.

6.5.1.1 Metadados de Projeto

Os Metadados de Projeto mantêm a mesma estrutura e as funcionalidades, originalmente propostas por [BEC06] e documentadas em [CUN05], de armazenar as informações que permitam: (i) o mapeamento entre o dado bruto e o dado requerido, e (ii) a localização e chamada dos *wrappers*.

Esses metadados possibilitam que variações na gestão de desenvolvimento de *software* que impliquem na extração de novas métricas ou variação das suas características (*e.g.* nome atributo, tabela de origem, domínio), possam ser efetivadas a partir da simples inclusão das mesmas, sem a necessidade de alterar a solução proposta para a Camada de Integração de Aplicações e, principalmente, com mínimo esforço de programação.

Para o desenvolvimento desta solução, os metadados foram estendidos para incorporar as métricas de monitoração, destacadas na Tabela 10, e informações de acesso às diferentes fontes de dados. Essas últimas são compostas pelos seguintes atributos: (i) URL da base de dados da ferramenta ou endereço físico do arquivo (documentos texto ou planilhas eletrônicas), (ii) *username* e (iii) *password* (Figura 21). Tanto a estrutura desses metadados, como o seu conteúdo utilizado nos testes (Capítulo 7), estão documentados no Apêndice B.

6.5.1.2 *Wrappers*

Os *Wrappers* são pacotes que encapsulam uma ou mais aplicações provendo uma única interface [ALO04]. Esta solução determina que cada uma das ferramentas de apoio à gestão do ambiente de desenvolvimento de *software* tenha um *wrapper* correspondente, responsável por esconder as suas especificidades e extrair as métricas armazenadas na sua base de dados. Além disso, os mesmos devem ser publicados e chamados conforme o padrão *WSDL,* descrito na Seção 2.5.3 e estão ilustrados na Figura 20 por semicírculos pretos.

A utilização de *wrappers*, segundo os padrões de *WS,* apresenta as seguintes vantagens: (i) possibilita a adoção de novas ferramentas e oferece suporte a novos projetos, unicamente com a definição e publicação de novos serviços e a criação de Metadados de Projeto; (ii) permite compartilhar código de serviços entre projetos que usem a mesma ferramenta, desde que essa apresenta estrutura de dados semelhante; (iii) permite a extração de dados entre diferentes ferramentas, executando em sistemas operacionais distintos; (iv) oferece liberdade de escolha de linguagem de programação para codificar os serviços.

Durante a execução desta pesquisa, foram definidos, implementados, publicados e testados *wrappers* de extração para as seguintes fontes: *MS Project*, *IBM Rational ClearQuest*

e planilhas *Excel*. Como a estrutura do documento *WSDL* e do protocolo *SOAP* são semelhantes para todos esses serviços, nesta seção é apresentada, somente, a solução adotada para o *MS Project.*

A Figura 21 mostra o documento *WSDL* gerado a partir do *wrapper* da ferramenta *MSProject*. Para facilitar o entendimento, o mesmo encontra-se dividido em parte abstrata e concreta, conforme a estrutura apresentada pela Figura 4, na Seção 2.5.3. Na primeira linha desse documento está definida a versão *XML* utilizada. Em seguida, os *xmlns* especificam a localização dos padrões de formatação empregados para criar o *WS* e o trecho targetNamespace=http://tempuri.org/ determina o *URI* (*Uniform Resource Identifier*) do servidor onde os serviços estão publicados.

A parte do documento *WSDL* delimitada pelo primeiro retângulo apresenta os elementos <*WSDL*:types> que definem o tipo dos dados trocados entre as mensagens *SOAP*, incluindo o número mínimo (minOccurs) e máximo (maxOccurs) de ocorrência dos mesmos. Com base nessa parte, percebe-se que o método **MSProject** apresenta tipo de dados complexo de entrada e de saída, sendo que ambos são *strings*, e a sua entrada é composta pelos atributos **url_EPM** (endereço da base de dados da ferramenta), **nomeprojeto** (projeto que deve ter suas métricas coletadas), **versão do projeto**, **username** e **password** (dados de acesso da base de dados da ferramenta). Por sua vez a saída apresenta um único parâmetro, denominado **MSProjectResult**, que retorna o resultado obtido após a execução do *WS*, contendo as métricas capturadas, concatenadas e separadas por um caractere especial.

A parte abstrata (assinalada na figura) é composta por mensagens, operações e *port types*. Assim, as *tags* <*WSDL*:message name="**MSProject*SOAP*In**"> e <*WSDL*:message name="**MSProject*SOAP*Out**"> informam, respectivamente, o nome das mensagens de chamada e de retorno do serviço. O elemento <*WSDL*:portType> apresenta o conjunto das operações de um determinado *WS*, e é considerado um elemento abstrato, pois não oferece informações sobre como se conectar diretamente a um *WS*. Cada uma dessas operações é definida pelo elemento *operation*, equivalente a uma chamada de método *Java* ou uma sub-rotina no *Visual Basic*. No caso do *wrapper* do MSProject, o mesmo apresenta somente a operação "**MSProject**", tendo como entrada e saída, respectivamente, as mensagens: "**tns:MSProject*SOAP*In**" e "**tns:MSProject*SOAP*Out**", declaradas anteriormente.

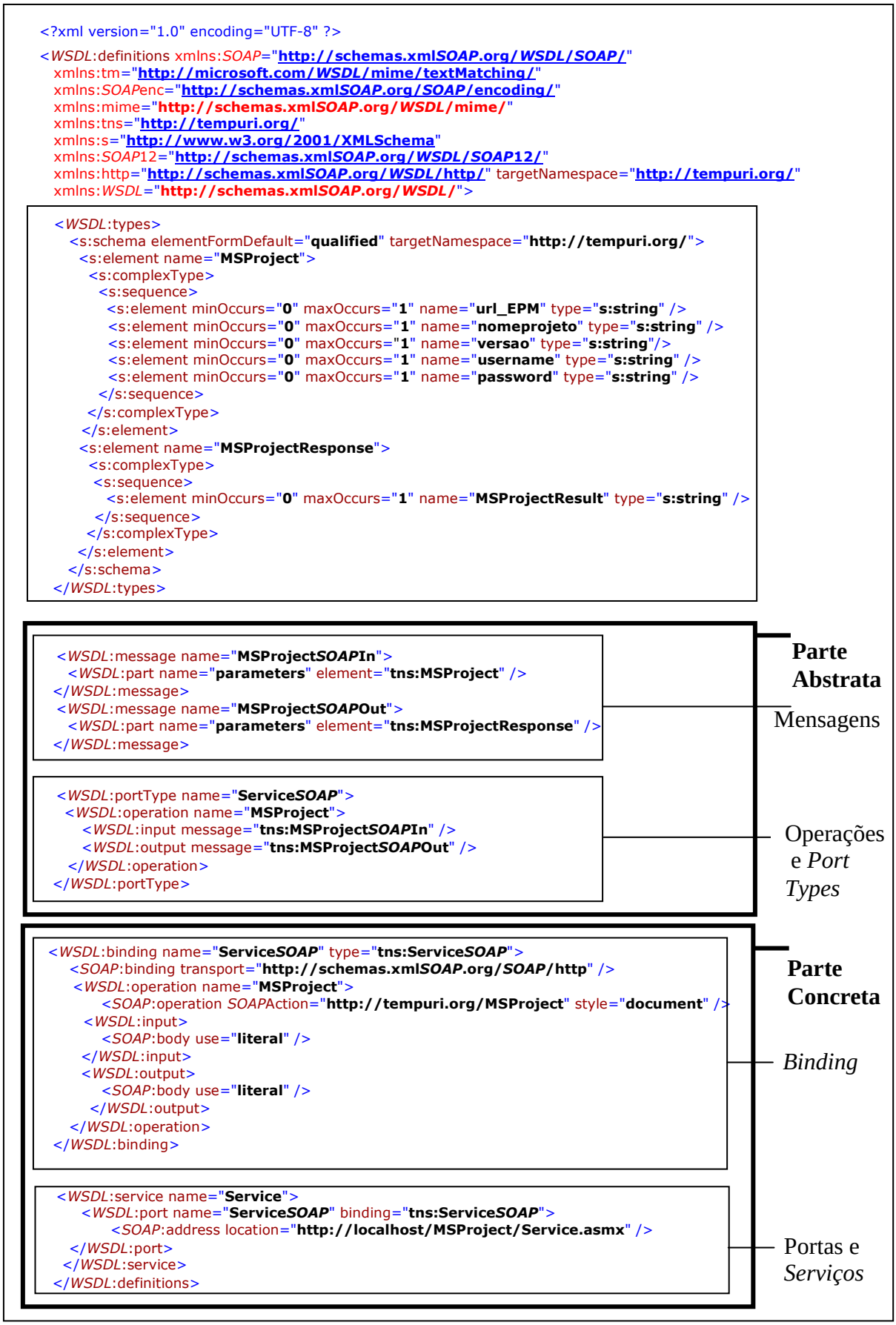


Figura 21 – *WSDL* do *MS Project*.

A parte concreta é composta por: *binding*, portas e serviços. O elemento *binding*, tomado como exemplo, é do tipo *SOAP* (Seção 2.5.2) e utiliza o protocolo de transporte *HTTP*. Já o elemento <*WSDL*:service> contém a definição, a localização física e o nome do serviço descrito no <*WSDL*:binding>.As Figura 22 e 23 ilustram, respectivamente, os protocolos *SOAP* de solicitação de um serviço "*Request*" e resposta "*Response*", responsáveis pela troca de mensagens entre os serviços.

```
POST /MSProject/Service.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/MSProject"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <MSProject xmlns="http://tempuri.org/">
      <url_EPM>string</url_EPM>
      <nomeprojeto>string</nomeprojeto>
      <username>string</username>
      <password>string</password>
    </MSProject>
  </soap:Body>
</soap:Envelope>
```

Figura 22 – *SOAP Request.*

```
http/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length:length
<?xml version="1.0" enconding=" utf-8"?>
<soap:Envelope
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope">
<soap:Body>
<MSProjectResponse xmlns="http://tempuri.org">
<MSProjectResult>string</MSProjectResult>
</MSProjectResponse>
</soap:Body>
</soap:Envelope>
```

Figura 23 – *SOAP Response.*

6.5.2 Rotina de Extração

A presença de um fluxo bidirecional de dados, e de camadas de serviços entre os *wrappers* e a Rotina de Extração, permite que o processo de ETC seja realizado de duas maneiras: sob demanda ou por iniciativa. Na primeira, a solução determina que a Rotina de Extração seja responsável pelo início da execução do processo, por intermédio ou de eventos pré-agendados ou da ação de um determinado recurso. Já na segunda, o processo pode ser

iniciado por *wrappers* pró-ativos, que realizam a chamada dessa Rotina a partir do ambiente do projeto, tanto quando valores atualizados estiverem disponíveis quando no momento em que os gestores acharem conveniente. Nesse sentido, as camadas de serviços permitem que o processo de ETC fique sempre à disposição para eventuais chamadas, estabelecendo a troca de conteúdo entre os projetos e o ambiente de *data warehousing*. Nas duas maneiras citadas, a Rotina de Extração efetua a chamada dos *wrappers*, com base nas informações contidas nos seus respectivos documentos *WSDL* e no Metadado de Projeto, tendo autonomia para se comunicar com a Camada de Integração de Dados, no momento em que achar adequado.

Assim, com base nos *wrappers*, nos Metadados dos Projetos e na Rotina de Extração, a coleta das métricas pode ser executada de modo automatizado, com o mínimo de intrusão e considerando um ambiente de desenvolvimento heterogêneo de ferramentas, modelos de processos e projetos. Essa automatização do processo de extração é fundamental para diminuir a latência e possibilitar uma alta freqüência de captura, essenciais à monitoração.

Durante a execução desta pesquisa, foi implementado um protótipo de Rotina de Extração, acionado através de uma interface *web* e contendo três passos distintos: 1) Localização e Leitura dos Metadados do Projeto; 2) Chamada dos *Wrapper*s especificados no metadados, passando como parâmetros as informações necessárias para capturar as métricas; e 3) Tratamento do Retorno dos *Wrappers* e Inserção das Métricas na *DSA*.

Os itens a seguir correspondem aos trechos de códigos desenvolvidos para efetuar os três passos citados, tomando como exemplo o *wrapper* do *MS Project*. É importante mencionar que esses trechos representam uma forma simplificada de implementar a Rotina de Extração, mas que a mesma pode apresentar variações, conforme as características do ambiente para o qual a solução está sendo aplicada.

- Passo 1: Localização e Leitura dos Metadados de Projeto

A leitura dos Metadados de Projetos (Figura 24) baseia-se na biblioteca System.Xml, que oferece métodos para acessar arquivos *XML*. A variável “doc”, do tipo XMLDocumento, faz referência ao conteúdo do arquivo contendo o metadado. O método “load” estabelece a relação entre essa variável e o documento *XML*, a partir da sua localização física, determinada pelo parâmetro “@caminho”. Tal variável deve conter o endereço do arquivo “.xml”, que representa o metadado, juntamente com o seu nome. A solução recomenda que os arquivos de Metadados de Projeto recebam o nome dos seus respectivos projetos, por exemplo: “Projeto1.xml”.

```
XmlDocument doc = new XmlDocument();
doc.Load(@caminho);
XmlNode node;

node = doc.SelectSingleNode("/Projeto/Ferramentas[Nome='MSProject']");
url = node.ChildNodes.Item(1).InnerText;
login = node.ChildNodes.Item(2).InnerText;
passwd = node.ChildNodes.Item(3).InnerText;
```

Figura 24 – Trecho referente ao Passo 1.

Os conteúdos dos metadados podem ser vistos como uma árvore, onde os diferentes nós contêm as informações desejadas. Assim, para acessá-los, utiliza-se o método "SelectSingleNode" que permite a seleção de um único nó com base nos valores passados como parâmetro. Por exemplo, a linha de comando `node = doc.SelectSingleNode("/Projeto/Ferramentas[Nome='MSProject']");` busca o nó denominado MSProject e armazena as suas informações na variável "node". Essa última pode ser vista, de maneira simplificada, como um vetor a partir do qual é possível acessar, por intermédio de índices, o conteúdo das suas posições através do método node.ChildNodes.Item().InnerText. Dessa forma, são instanciadas as variáveis utilizadas como parâmetros de entrada durante a chamada dos *wrappers*.

- Passo 2: Chamada dos Wrappers

A chamada dos *wrappers* é realizada através de métodos da biblioteca System.Web. As primeiras linhas do trecho ilustrado (Figura 25) mostram a criação do serviço "servicoMSP", onde está declarado o *wrapper* MSProject, recebendo como *namespace* "localhostMSProject". A última linha constitui a chamada do método denominado MSProject, oferecido pelo "servicoMSP", com seus respectivos parâmetros de entrada (`url,nomeprojeto,username,passwd`). O resultado dessa execução fica armazenado na variável retorno.

```
localhostMSProject.Service servicoMSP = new
localhostMSProject.Service();
retorno = servicoMSP.MSProject(url,nomeprojeto,username,passwd);
```

Figura 25 – Trecho referente ao Passo 2.

- Passo 3: Tratamento do Retorno do Wrappers e Inserção das Métricas na DSA

O conteúdo do retorno dos serviços deve ser interpretado, conforme a estrutura e as informações armazenadas no Metadado de Projeto, e enviado à *DSA*. Na solução codificada durante a execução desta pesquisa, adotou-se que os *wrappers* retornam as métricas por intermédio de uma *string* concatenada, que inclui caracteres especiais de separação. O trecho ilustrado pela Figura 26 apresenta: (i) a declaração das variáveis, (ii) a *string* para conexão, (iii) os comandos de conexão, que permitem acessar a *DSA*, (iv) a interpretação da *string* de retorno do *wrapper*, e (v) a inserção das métricas em uma tabela temporária (DSA_Produto), presente na *DSA*.

```
OleDbConnection objConnection = null;
OleDbCommand objCmd1 = null;
OleDbCommand objCmd2 = null;

string strSQL;
string strConnection = "Data Source=(local);User
ID=hpsa;Password=hpsasa; Initial Catalog=BaseAnalitica;
Provider=SQLNCLI.1; Persist Security Info=True; Auto Translate=False;";

objConnection = new OleDbConnection(strConnection);
objConnection.ConnectionString = strConnection;
objConnection.Open();

string[] blocos = retorno.Split('*');
string[] linhas = blocos[0].Split('@');
string[] dados;

int i = 0;
for (i = 1; i < linhas.Length; i++){
    teste = linhas[i];
    dados = linhas[i].Split(';');
    strSQL = "INSERT into DSA_Produto values(" + dados[0] + ",'" +
    dados[1] + "','" + dados[2] + "','" + dados[3] + "'," + dados[4]
    + "," + dados[5] + "," + dados[6] + "," + dados[7] + ",'" + dados[8]
    + "','" + dados[9] + "','" + dados[10] + "','" + dados[11] + "','" +
    dados[12] + "'," + dados[13] + ",0," + dados[15] + "," + dados[16] +
    ",'" +dados[17] +  "',01/05/2007," + dados[18] + "," + dados[19]+)";
    TxtRetorno.Text = strSQL;
    try{
      objCmd2 = new OleDbCommand(strSQL, objConnection);
      objCmd2.ExecuteNonQuery();
   }catch{
      TxtRetorno.Text = "Erro no INSERT TMP_Produto";
   }
}
```

Figura 26 – Trecho referente ao Passo 3.

6.5.3 Camada de Integração dos Dados

A Camada de Integração de Dados é responsável pela transformação e limpeza dos dados armazenados temporariamente na *DSA*. Esta seção descreve os principais aspectos relacionados com: (i) a determinação das Rotinas de Limpeza e Transformação, e (ii) a estrutura e construção da *DSA*. Com relação ao Metadado Organizacional, adotou-se nesta pesquisa a mesma solução proposta e documentada em [CUN05].

6.5.3.1 *Data Staging Area DSA*

Visando minimizar a intrusão do processo de ETC das métricas, utiliza-se uma área de temporária de dados, denominada *DSA*, composta por um conjunto de tabelas onde os dados brutos extraídos das fontes de origem devem ser armazenados e pré-processados por intermédio das Rotinas de Limpeza e Transformação (Seção 6.5.3.2), considerando o modelo analítico alvo descrito na Seção 6.3. A solução proposta tem como fonte de dados as ferramentas *MS Project*, *ClearQuest* e planilhas *MS Excel*. Porém entende-se que esses passos podem ser seguidos independentemente das fontes adotadas.

Estrutura da DSA

A solução proposta e implementada para estruturar a *DSA* segue os passos sugeridos por [KIM98] e listados na Seção 2.4.3.3. Os itens a seguir relatam as ações e os resultados obtidos a partir da execução desses.

Passo 1: determina a criação de um plano de alto nível, contendo as fontes de origem dos dados, as transformações requeridas e a carga nas tabelas alvo. Como resultado obteve-se o plano ilustrado pela Figura 27, a partir do qual foi possível obter uma visão geral do processo de ETC.

Passo2: aconselha implementar uma solução dedicada quando a natureza do ambiente de desenvolvimento apresenta sistemas de dados personalizados, baseados em banco de dados proprietários. Com base nas características do ambiente heterogêneo, optou-se por desenvolver uma *DSA* dedicada. Para tanto, foi definida a estrutura das tabelas temporárias, incluindo seus atributos e domínios, considerando o modelo analítico proposto (Seção 6.3). A mesma encontra-se no Apêndice C.

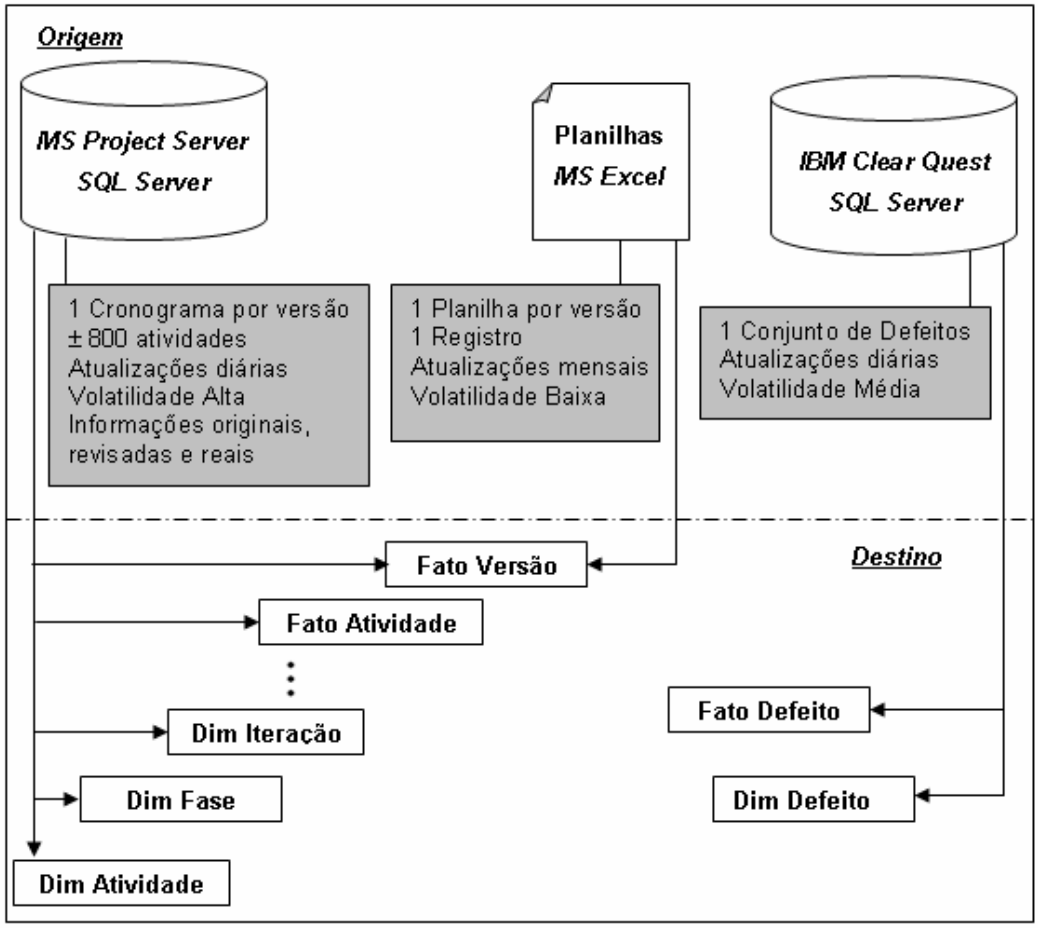


Figura 27 – Plano Alto Nível referente ao Passo 1.

Passo 3: determina a elaboração de planos detalhados para cada uma das fontes de dados, contidas no plano de alto nível. Como resultados da sua execução foram criados os planos ilustrados pelas Figuras 28 e 29 que correspondem, respectivamente, à base de dados do *MS Project* e do *ClearQuest*. O plano referente à Planilha *MS Excel* também foi efetuado. Em ambos os planos são apresentadas as tabelas que contêm os dados brutos, juntamente com a especificação, em alto nível, da consulta que deve ser realizada para extrair os dados que compõem primeiramente as tabelas temporárias (*e.g.* DSA_Atividade e DSA_Defeito) e, posteriormente, as fatos e as dimensões. Esses planos também ilustram as limpezas e transformações inicialmente realizadas (*e.g.* ntext→varchar) para que os dados brutos possam ser devidamente armazenados no DW, conforme a estrutura proposta pelo modelo analítico (Fato_Atividade, Fato_Versão e Fato_Defeito e as dimensões). Dessa forma, foi possível obter uma visão inicial completa de todas as etapas que envolvem o processo de ETC dos dados.

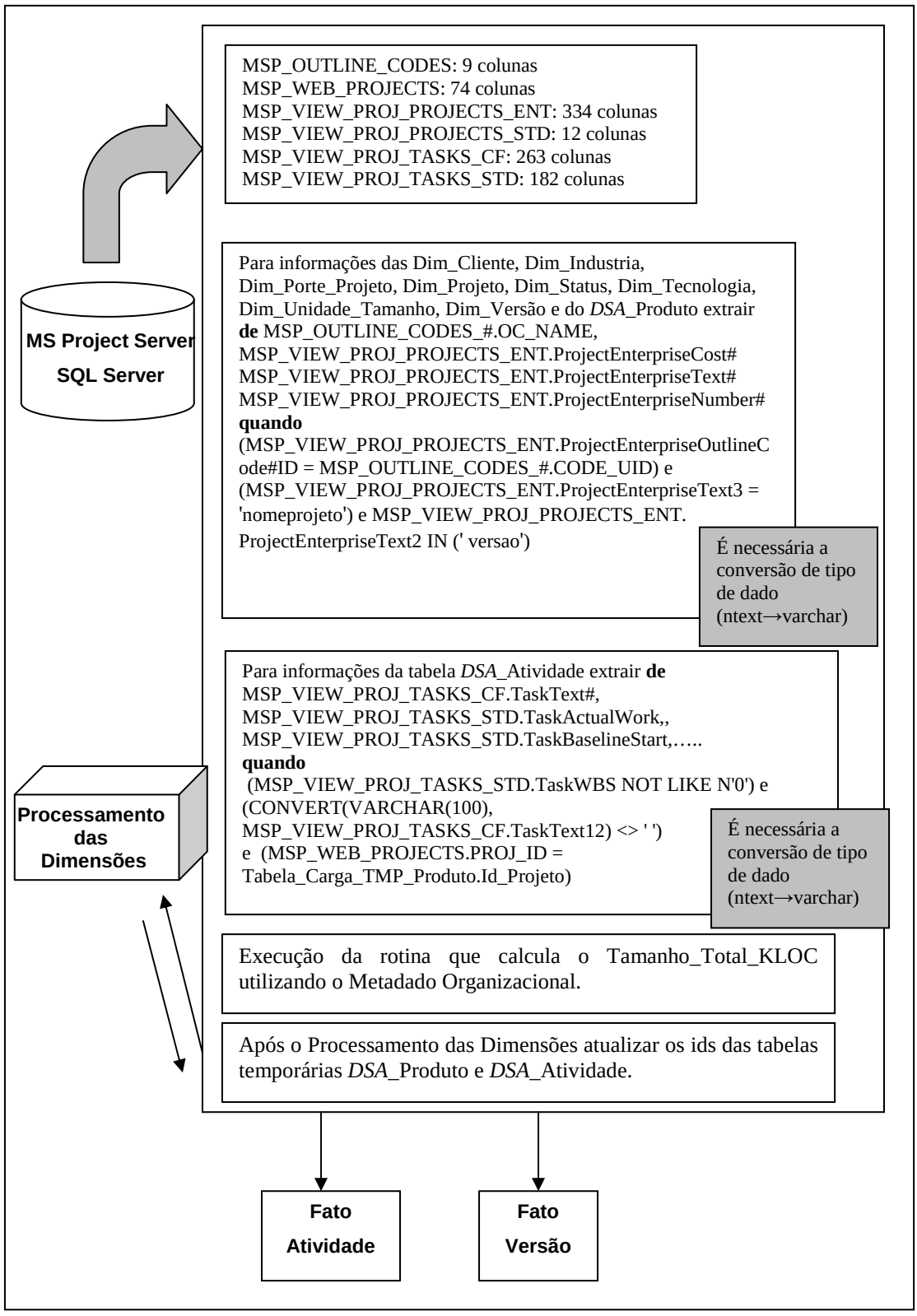


Figura 28 – Plano Detalhando do *MS Project.*

Passo 4: determina, por questões de facilidade, que as tabelas dimensão sejam populadas primeiro, e que suas chaves sejam atualizadas a cada nova linha incluída, sendo definida uma única chave de acesso por registro. Para tanto, esta solução

determina que sempre que um novo conjunto de métricas for extraído a rotina responsável por atualizar as dimensões deve ser executada (Rotina 2 - Atualizar Dimensões, Seção 6.5.3.2).

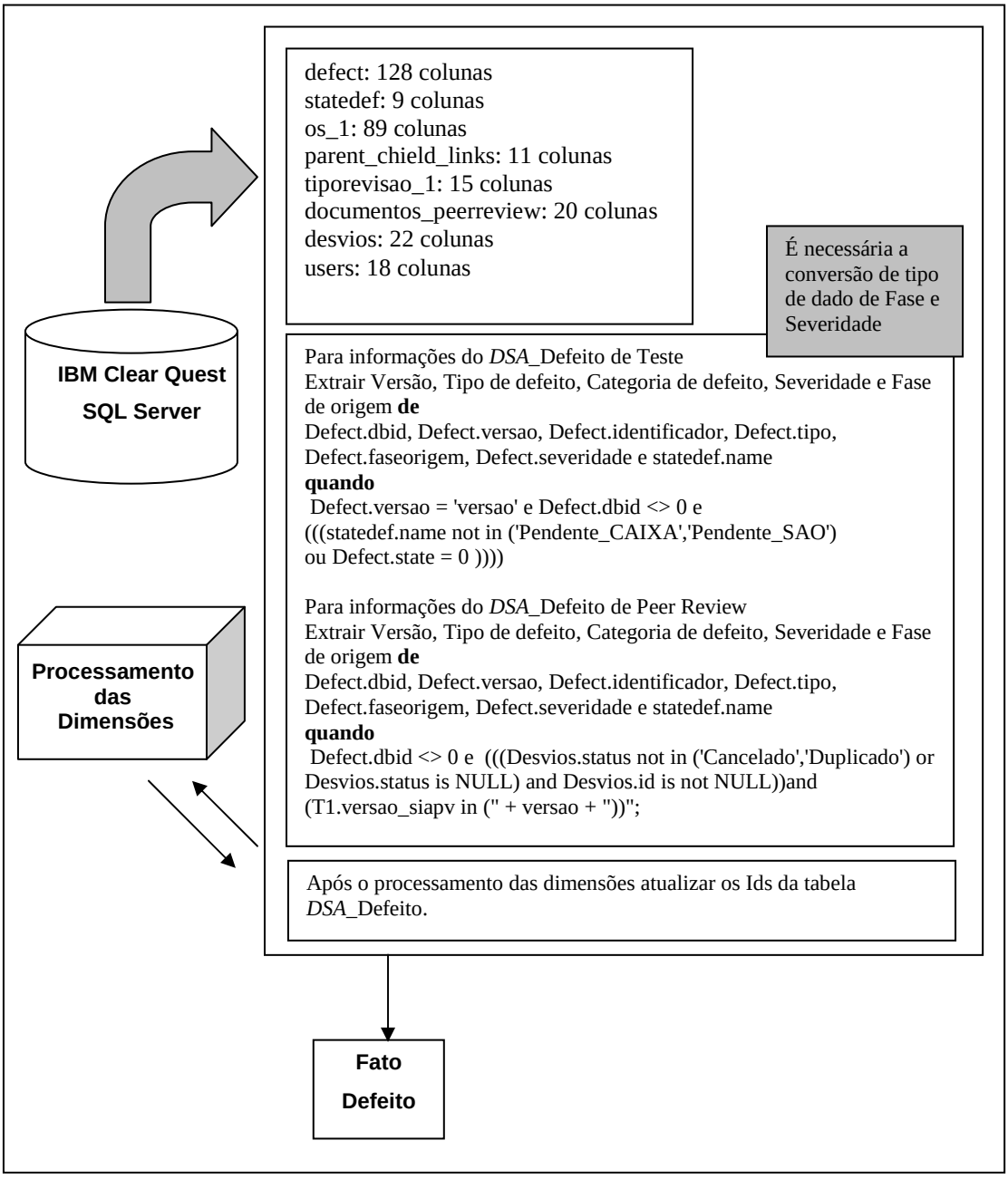


Figura 29 – Plano Detalhado do *ClearQuest.*

Passo 5: determina a realização dos testes para validar o conteúdo das dimensões e as atualizações das suas chaves. Os mesmos foram realizados durante o desenvolvimento da solução e sua execução pode ser considerada pré-requisito para a realização dos testes relatados no Capítulo 7.

Passo 6: determina métodos para atualizar valores das dimensões existentes no DW. Por questões de conveniência e limitação de tempo esse passo não foi seguido, esta solução define que os atributos das dimensões não podem ter seu conteúdo atualizado. Por essa razão, quando novos valores de dimensões forem necessários, os mesmos devem ser inseridos como novos registros.

Passos 7, 8 e 9: estão relacionados com a alimentação, atualização e teste do conteúdo das tabelas fato. Esses passos foram seguidos, a partir da Rotina 3 – Atualiza Fatos, definida na Seção 6.5.3.2.

Passo 10: propõe a automação da aplicação de *staging.* Para tanto, esta solução definiu e implementou serviços *SSIS* contendo componentes especiais que efetuam as rotinas de Limpeza e Transformação apontadas pelos passos 4 e 8, escritas em *SQL* e especificadas na Seção 6.5.3.2. Neste passo, também foi definida e implementada a rotina de carga incremental, descrita na Seção 6.5.4 e um ambiente de cubo *OLAP*. Esse último foi desenvolvido a partir da ferramenta de *BI* do *MS Visual Studio*, para testar a carga incremental e o conteúdo das tabelas que compõem o *DW*, com base nas informações temporariamente armazenadas na *DSA*.

6.5.3.2 Rotinas de Limpeza e Transformação

A definição das Rotinas de Limpeza e Transformação adotadas por esta solução segue algumas das atividades de transformação propostas por [KIM98], descritas na Seção 2.4.3.2, sendo elas:

- resolução de domínios, realizada pelos *wrappers* durante a extração das métricas, para solucionar domínios conflitantes, dados faltantes e formatar valores conforme o padrão pré-estabelecido, a partir das informações contidas no Metadado Organizacional (*e.g.* padronização da severidade de defeitos);
- combinação das fontes de dados, verificando a integridade entre as chaves primárias (*e.g.* mapeando as chaves utilizadas na base de dados de origem das ferramentas para as chaves das dimensões);
- determinação de chaves para cada registro das dimensões, com a finalidade de garantir as dependências entre os dados.

As demais não foram consideradas, pois não existem ruídos e dados inúteis, já que somente dados requeridos são extraídos da origem.

A solução implementada, em *SQL* e disponibilizada por *SSIS*, apresenta as seguintes rotinas: 1) Inicializar; 2)Atualizar Dimensões; e 3)Atualizar Fatos. A primeira alimenta a Dim_Tipo_Fato com os seguintes valores: original, revisado e real. A segunda realiza a leitura das tabelas temporárias e verifica se os seus valores existem nas tabelas dimensões (Tabela 14), caso essa condição não seja satisfeita, efetua a sua inclusão. A terceira realiza a leitura de todas as tabelas temporárias e, a partir de agrupamentos e condições, gera as tabelas fato (Tabela 13), juntamente com as chaves estrangeiras das dimensões que caracterizam o fato.

6.5.4 Carga do Repositório de Dados

Para oferecer suporte adequado à monitoração e manter um Repositório de Dados atualizado e consistente, comportando cargas freqüentes, esta solução propõe um procedimento de carga incremental, conforme os conceitos e métodos definidos por [KIM98], descritos na Seção 2.4.3.3. A partir do mesmo, é possível manter o histórico e o acompanhamento regular do andamento do projeto. Para tanto, a carga incremental considera a validade dos fatos, segundo a definição apresentada na Seção 6.3.3.

Com base nessas definições e conceitos relacionados com a validade dos fatos, foram criadas três regras que regem a carga incremental que têm, como objetivo, controlar inclusões, atualizações e término da validade dos dados.

- **Regra 1**: quando um novo fato apresentar chave primária diferente da previamente armazenada no *DW*, o mesmo deve ser inserido recebendo como D_I a data da carga e D_F o valor nulo.
- **Regra 2**: quando um fato atualizado, caracterizado por apresentar chave primária igual à previamente armazenada no *DW*, com exceção do valor que corresponde à data de carga, e pelo menos uma das suas métricas com valor diferente, o mesmo deve ser inserido no *DW*, recebendo como D_I a data da carga e D_F o valor nulo.
- **Regra 3**: os fatos semelhantes (mesma chave primária, com exceção do identificador de data de validade, e contendo pelo menos uma métrica diferente e com data de validade final (D_F) nula) aos inseridos pela regra 2 devem ter seu período de validade atualizado. Para tanto, o seu valor de D_F recebe a data corrente e o seu D_I permanece inalterado.

Durante a execução desta pesquisa, foi implementado um procedimento de carga, contendo as três regras citadas anteriormente, escritas em *SQL*¸ e disponibilizado na forma de

um *WS*. Por questões de conveniência, essa carga é chamada por um serviço *SSIS* do *MS SQL Server*. Assim, depois que os dados encontram-se devidamente estruturados e consolidados na *DSA* esse procedimento é executado tendo como resultado os dados corretamente armazenados no *DW*.

Esta solução determina que a carga no *DW* seja realizada por intermédio de serviços, ilustrados na segunda camada da arquitetura do *SPDW+* (Figura 20). Dentre os benefícios dessa forma de realização, pode-se facilmente citar: (i) substituir o ambiente da base de dados que armazena o Repositório de Dados; (ii) utilizar os mesmos serviços de carga em ambientes semelhantes; e (iii) inserir novas métricas no procedimento, a partir da publicação de serviços.

6.6 Considerações

A solução descrita neste capítulo, denominada *SPDW+*, propõe um processo automatizado de ETC de métricas de *software* a serem armazenadas em um Repositório de Dados (*DW*), na forma de um ambiente de *data warehousing*. As células da Tabela 15 contêm uma tabulação dos domínios dos aspectos envolvidos pela automação computacional, levantados na Seção 3.1, considerados necessários para mensurar a qualidade de *software*. A solução proposta trata tanto o nível organizacional, como o de projeto, considerando atividades de análise e monitoração. Para facilitar o entendimento e sistematizar a explicação, os próximos itens descrevem como a solução proposta para tratar os aspectos da plataforma computacional, resolve o problema. Por questões de simplificação esses itens endereçam o nível organizacional que, quando atendido, também resolve as questões relacionadas ao nível de projeto.

Latência e Freqüência: para tratar esses dois aspectos a solução apresenta um processo automatizado de ETC das métricas orientado a serviços. O mesmo permite que a coleta das métricas seja realizada por demanda ou por iniciativa, tão logo novos valores forem lançados, pelos recursos, nas ferramentas usadas no projeto para acompanhamento, ou conforme as políticas de coleta estabelecidas pela organização. Assim, é possível ter uma freqüência e uma latência com ordem de grandeza de horas, considerada essencial à monitoração.

Heterogeneidade de Modelos de Processos: para endereçar de maneira adequada esse aspecto, a solução propõe um modelo analítico abrangente e flexível, que permite que os diferentes projetos da organização adotem modelos de processos distintos (*e.g.* cascata,

cascata com sobreposição, iterativo, evolutivo e unificado). A estrutura das tabelas fatos e dimensões apresentam métricas aditivas e semi-aditivas que possibilitam diferentes perspectivas de análise e níveis de sumarização, conforme as etapas dos ciclos de vida adotados pelos modelos de processo.

Tabela 15 – Tabela de interações tratadas pelo *SPDW+*.

	Atividades de Supervisão				
Nível de Mensuração	**Análise**	**Monitoração**	**Previsão**	**Plataforma Computacional**	
De Projeto Organizacional	Horas (√)	Horas (√)	-	**Latência**	
De Projeto Organizacional	Horas (√)	Horas (√)	-	**Freqüência**	
De Projeto	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	-	**Modelos de Processo**	**Heterogeneidade**
	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	-	**Ferramentas**	
	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	-	**Projetos**	
	Compartilhado(S)	Compartilhado(S)	-	**Nível de Isolamento**	
Organizacional	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	-	**Modelos de Processo**	
	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	-	**Ferramentas**	
	Homogêneo (S) Heterogêneo (S)	Homogêneo (S) Heterogêneo (S)	-	**Projetos**	
	Compartilhado(S)	Compartilhado(S)	-	**Nível de Isolamento**	
De Projeto Organizacional	Isento (√)	Isento (√)	-	**Intrusão**	

Heterogeneidade de Ferramentas: para resolver esse aspecto a solução propõe a utilização de *wrappers* de extração, que escondem as especificidades das ferramentas e Metadados de Projeto, que são responsáveis pelo mapeamento entre cada dado requerido e o correspondente dado bruto. Dessa forma, as métricas podem ser extraídas de diferentes fontes, necessitando apenas da publicação dos respectivos serviços de extração. Além disso, a solução permite que novas ferramentas ou versões das mesmas possam ser inseridas no ambiente, somente com a publicação de novos serviços. Novos projetos também podem se beneficiar desses *wrappers*, pois os *wrappers* podem ser adotados por projetos distintos, desde que esses adotem as mesmas ferramentas, para as mesmas métricas.

Heterogeneidade de Projetos: a solução proposta trata adequadamente a heterogeneidade de projetos, permitindo que os mesmos sejam de pequeno ou grande porte. Para tanto, são utilizados *wrappers*, que permitem tanto a extração de ferramentas de apoio à gestão do ambiente de desenvolvimento de *software*, como de planilhas *MS Excel*.

Heterogeneidade de Nível de Isolamento: a solução resolve apenas para o nível de isolamento compartilhado, não tratando redes isoladas que, necessariamente, dependem da intervenção humana para serem suportadas.

Intrusão: a solução propõe um processo automatizado de ETC das métricas, que pode ser iniciado sob demanda ou por iniciativa, baseado em serviços. Dessa forma, é possível coletar as métricas e disponibilizar seus resultados sem que seja necessário haver envolvimento dos recursos do projeto, caracterizando o processo como isento de intrusão.

Com base nos aspectos tratados pela solução, pode-se afirmar que o *SPDW+* oferece suporte: a análise e monitoração da mensuração da qualidade de *software*, nos dois níveis, de projeto e organizacional, considerando todas as heterogeneidades, com exceção do nível de isolamento, e apresenta um nível isento de intrusão. Além disso, pode-se afirmar que, devido à flexibilidade e abrangência do modelo analítico, a aplicação do mesmo é possível em organizações de desenvolvimento de *software*, que apresentem estrutura de projeto semelhante, exigindo, apenas, a realização de pequenas modificações.

A partir da comparação entre esta solução e a original (*SPDW*), é possível perceber que a primeira propõe um modelo analítico mais abrangente e mais elegante, com suporte à análise e monitoração, conseqüente de um Programa de Métricas estendido, segundo a técnica *EVA*, e de uma estrutura de projetos estendida. Além disso, esta solução possibilita a captura de métricas no grão de atividade, aumentando as perspectivas de análise, garantindo um acompanhamento regular deste nível. Isso tudo, com um número menor de tabelas, contendo apenas métricas aditivas e semi-aditivas, suportando todos os níveis de sumarização. Esta solução também apresenta definições mais detalhadas dos *wrappers* e das rotinas (extração e limpeza e transformação). Ainda, cabe salientar que esta pesquisa implementou todos os componentes relatados neste capítulo, apresentando uma solução completa, desde a obtenção das métricas até a disponibilização dos seus resultados, a partir de uma interface de cubo *OLAP*.

7 TESTE DA SOLUÇÃO

Este capítulo relata o conjunto de testes realizados sobre a solução proposta no Capítulo 6. Para tanto, apresenta: (i) os objetivos e procedimentos de teste; (ii) os recursos computacionais, configurados para criar o ambiente de teste em laboratório; (iii) as fontes de dados utilizadas; (iv) a estrutura do cubo *OLAP* construído, utilizado como interface de teste; e (v) os resultados obtidos e a análise dos mesmos.

7.1 Objetivos

Os testes têm como objetivo principal avaliar se o processo automatizado de ETC proposto pelo *SPDW+* consegue tratar adequadamente, e em tempo compatível, os aspectos que envolvem a mensuração da qualidade de *software*, segundo a Tabela 15. Tratar adequadamente significa propiciar pelo menos as funcionalidades oferecidas para análise propostas por [CUN05], e mais as facilidades para monitoração. Em tempo compatível significa que a latência dos dados e a freqüência de coleta devem ser, no máximo, da ordem de horas.

Nesse sentido, foi criado um ambiente de testes com a implementação do modelo analítico proposto (Seção 6.3) e dos componentes das camadas de Integração de Dados e de Aplicações (Figura 20): (i) *wrappers* para *MS Project*, *ClearQuest* e planilhas *MS Excel*, (ii) Rotina de Extração, (iii) Rotinas de Limpeza e Transformação e (iv) *DSA*. O ambiente de teste é de baixa intrusão em sua concepção, por necessitar de intervenção humana apenas para ativar alguns passos do processo.

Esse ambiente de testes apresenta, como única característica heterogênea, o uso de distintas ferramentas de armazenamento de métricas. O mesmo não trata heterogeneidade de modelos de processo de *software*, por já ter sido tratada em [CUN05], de tipos de projetos e de níveis de isolamento. Os testes realizados utilizam cronogramas consistentes com projetos reais de grande porte (Seção 3.1.3.2), como forma de testar tipos de projetos, e foram efetivados em um único computador (portanto, sem isolamento).

Dois tipos de teste foram efetuados, com: (1) versões encerradas (análise) e (2) versões em andamento (monitoração). No primeiro tipo, foram criados cinco cronogramas com versões encerradas (**%TC** igual a 100%). No segundo, para emular um acompanhamento regular das atividades de uma única versão, ao longo de três dias, foram construídos três cronogramas dessa versão, com pequenas alterações diárias. Os testes consistem,

basicamente, na execução do processo de ETC das métricas, a atualização do modelo analítico, a geração do cubo de dados, e a verificação dos valores das métricas com o auxílio de tabelas pivotantes.

A execução desse processo de ETC é composta pelos seguintes passos, a partir da publicação de um cronograma: (i) execução da Rotina de Extração (Seção 6.5.2), escolhendo o projeto e a versão desejada; (ii) execução do *SSIS* que contém as Rotinas de Limpeza e Transformação (Seção 6.5.3.2); (iii) chamada do *WS* responsável pela carga incremental no modelo analítico (Seção 6.5.4); e (iv) atualização dos valores do cubo *OLAP*.

Após a execução dos passos que compõem o processo de ETC, foi realizada uma série de consultas no cubo, como auxílio de tabelas pivotantes, para verificar se o *SPDW+* atende os seguintes objetivos específicos:

1) oferece suporte à análise e à monitoração da mensuração da qualidade de *software*, considerando: perspectivas de análise e níveis de sumarização distintos;
2) permite a monitoração segundo as métricas de *EVA* incorporadas no programa de métricas (Seção 6.2.1);
3) apresenta um processo automatizado ETC, a partir: das rotinas de Extração e Limpeza e Transformação, dos *wrappers* e da carga incremental.

7.2 Descrição dos Recursos Computacionais do Ambiente de Teste

Para a realização dos testes foi construído um ambiente de laboratório, nas dependências do GPIN, onde foram instalados e, em alguns casos configurados, os seguintes recursos computacionais:

1. IIS 6.0: necessário para publicar os serviços (*wrappers* e a carga incremental) e permitir que os mesmos possam ser acessados pelo *browser ou* chamados pelos componentes especiais para *WS* existentes nos serviços *SSIS*, do *SQL Server* 2005.

2. MS Project 2003: utilizado para construir cronogramas contendo as métricas diretas das áreas de qualidade: Tempo, Esforço, Custo, Tamanho e Qualidade (Satisfação do Cliente), que compõem o Programa de Métricas proposto por esta solução (Tabela 10).

3. *MS Project Server*: instalado e configurado para permitir que as informações dos cronogramas, possam ser armazenadas em uma base de dados estruturada, denominada *EPM*, localizada no *SQL Server*. Esse aplicativo apresenta uma

interface *web* para gerenciar projetos, a partir da qual é possível publicar os cronogramas e salvar seu conteúdo no *EPM*. O *Project Server* também permite a configuração de campos especiais (*Enterprise Custom Fields*), responsáveis por habilitar a personalização do cronograma conforme as necessidades da organização. Nesta solução foram adicionados campos desse tipo para manter informações das áreas de qualidade Tamanho e Custo, além de dados referentes ao modelo de processo adotado pelo projeto (*e.g.*versão, fase, iteração). Durante a instalação desse aplicativo foi necessário realizar uma série de configurações e executar alguns *scripts* adicionais para que o *Project Server* conseguisse acessar corretamente a nova versão do *SQL Server* 2005.

4. *MS Visual Studio* 2005: instalado e empregado para codificar os *wrappers* em J# e a Rotina de Extração em C#, que apresenta bibliotecas especiais para: tratar documentos *XML* (‘System.Xml’), realizar conexões com bases de dados (‘System.Data.OleDb’) e construir *WS* (‘System.Web’).
5. *MS SQL Server* 2005 *Professional*: instalado e utilizado como sistema gerenciador de banco de dados, responsável por armazenar as soluções propostas para o Repositório de Dados (*DW*), a *DSA*, bem como as bases de dados do *EPM* e do *ClearQuest.*
6. *MS Server Business Intelligence Development Studio*: aplicado na construção do cubo *OLAP* e dos serviços *SSIS* que contêm as Rotinas de Limpeza e Transformação. Este aplicativo também é responsável pelo cálculo das métricas derivadas, a partir de *scripts MDX* (*Multidimensional Expressions*), e pela visualização do conteúdo do cubo por intermédio de tabelas pivotantes.

7.3 Descrição dos Dados do Ambiente de Teste

Os dados utilizados nos testes são fictícios, mas representam situações bem próximas das reais, e têm por base a estrutura e as ferramentas adotadas no cenário real, relatado no Capítulo 5. Para tanto, o ambiente de teste conta com as seguintes fontes de dados: (i) Planilhas *MS Excel*; (ii) base de dados do *ClearQuest*; e (iii) base de dados do *EPM.*

7.3.1 Planilhas *MS Excel*

As planilhas *MS Excel* são constituídas por métricas diretas da área de qualidade Requisitos (**NMA**, **NMR**, **NRM** e **NRE**), nome do projeto e nome da versão, conforme a estrutura ilustrada pela Figura 30.

Projeto	Versão	NMA	NMR	MRM	NRE
Projeto1	01.00.00	24	19	6	0

Figura 30 – Conteúdo e Estrutura das Planilhas *MS Excel.*

7.3.2 Base de Dados do *ClearQuest*

A base de dados do *ClearQuest* armazena nas suas tabelas, entre outras informações, os seguintes dados: nome projeto, nome da versão, fase de origem do defeito, grau de severidade (A – alto, M – médio e B - baixo), tipo do defeito (interno e externo) e quantidade. A partir dessas informações é possível extrair as métricas diretas de Qualidade **NDI** e **NDE**. A Figura 31 ilustra o resultado de uma consulta contendo: o nome do projeto, a versão, o identificador do defeito, o seu tipo (interna- interno e homologação - externo), a fase de origem (fases do ciclo de vida do projeto) e o seu grau de severidade. Além desses dados, também são extraídos data de abertura e fechamento do defeito e iteração.

Results | Messages

	projeto	versão	identificador	tipo	faseorigem	severidade
27	Projeto1	06.00.00	Inc-9198	Homologação	Projeto	3
28	Projeto1	06.00.00	Inc-9198	Homologação	Projeto	3
29	Projeto1	06.00.00	DI-2430	Interna	Projeto	3
30	Projeto1	06.00.00	DI-2430	Interna	Projeto	3

Query executed successfully.

Figura 31 - Estrutura de Dados do *ClearQuest.*

7.3.3 Base de Dados do *EPM*

A base de dados do *EPM* mantém métricas das seguintes áreas de qualidade: Tempo, Esforço, Custo, Tamanho e Qualidade (somente o **ISC**), obtidas a partir da publicação de cronogramas. Esses últimos são estruturados conforme a hierarquia da Figura 32, proposta originalmente por [CUN05]. Dessa forma, é possível planificar os modelos de processo

suportado pela solução (cascata, iterativo, evolutivo e unificado), descritos na Seção 2.1.1, a partir de seis níveis distintos.

Para testar a solução foram criados e publicados oito cronogramas. Os cinco primeiros correspondem a versões encerradas (01.00.00 até 05.00.00), contendo todas as suas atividades concluídas (**%TC** igual a 100%), e que, por essa razão, não podem ser monitoradas. Os restantes simulam uma única versão em andamento (06.00.00), durante três datas distintas (04/06/2007, 05/06/2007 e 06/06/2007), com atividades: não iniciadas, em andamento e encerradas.

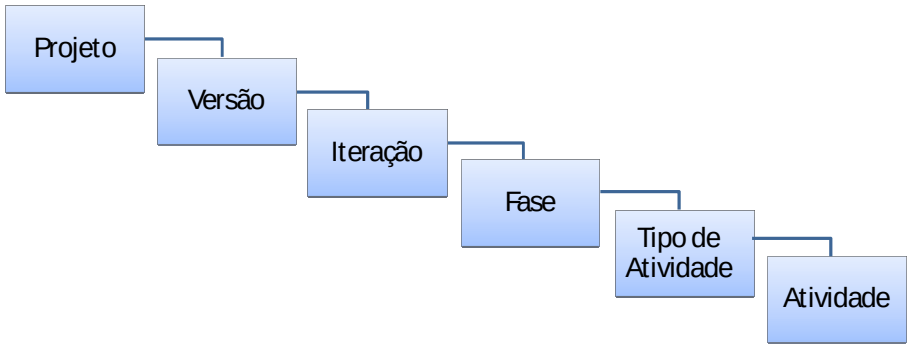


Figura 32 – Hierarquia dos cronogramas.

A Figura 33 mostra, em uma forma reduzida, o cronograma da versão 06.00.00, onde são listadas as colunas do *baseline* original. Vale ressaltar que as mesmas também existem para dados reais e do *baseline* revisado, mas foram suprimidas na figura. As linhas do cronograma representam os níveis de versão, fase, tipo de atividade e atividade. Os dados mostrados são relativos ao terceiro dia de acompanhamento do cronograma (05/06/2007), das atividades (DES – OS1...OS3) do tipo Trabalho, pertencentes a fase de Design, da versão 06.00.00, do Projeto 1. As colunas correspondem às métricas das áreas de qualidade: Tempo, Esforço e Custo, que compõem o Programa de Métricas (Tabela 10) . Cabe salientar que a utilização de cronograma, e de uma ferramenta dedicada para a sua construção, espelha o ambiente de um projeto de grande porte (Seção 3.1.3.2).

Além da estrutura de cronograma padrão usada, foram criados, no *Project*, dados personalizados do projeto (*Enterprise Custom Fields*), a partir de uma tela especial (Figura 34). Foram definidas as métricas da área de qualidade Tamanho e Qualidade (ISC), além de informações adicionais relacionadas com as Dimensão de Projeto (Tabela 14) e com o nível de iteração.

Task Name	% Work Complete	Esforço - BO	Data Inicial - BO	Data Final - BO	Custo - BO
Projeto1 06.00.00	**45%**	**322,4 hrs**	**Fri 1/6/07**	**Fri 22/6/07**	**4.492,80**
Design	**45%**	**78,5 hrs**	**Fri 1/6/07**	**Wed 6/6/07**	**942,00**
Trabalho	**45%**	**52 hrs**	**Fri 1/6/07**	**Tue 5/6/07**	**624,00**
DES - OS1	50%	17 hrs	Fri 1/6/07	Mon 4/6/07	204,00
DES - OS2	50%	14 hrs	Fri 1/6/07	Mon 4/6/07	168,00
DES - OS3	33%	21 hrs	Fri 1/6/07	Tue 5/6/07	252,00
Revisão	**0%**	**16 hrs**	**Mon 4/6/07**	**Tue 5/6/07**	**192,00**
DES - OS1	0%	4 hrs	Mon 4/6/07	**Mon 4/6/07**	48,00
DES - OS2	0%	7 hrs	Mon 4/6/07	**Tue 5/6/07**	84,00

Figura 33 - Cronograma da versão 06.00.00 em andamento.

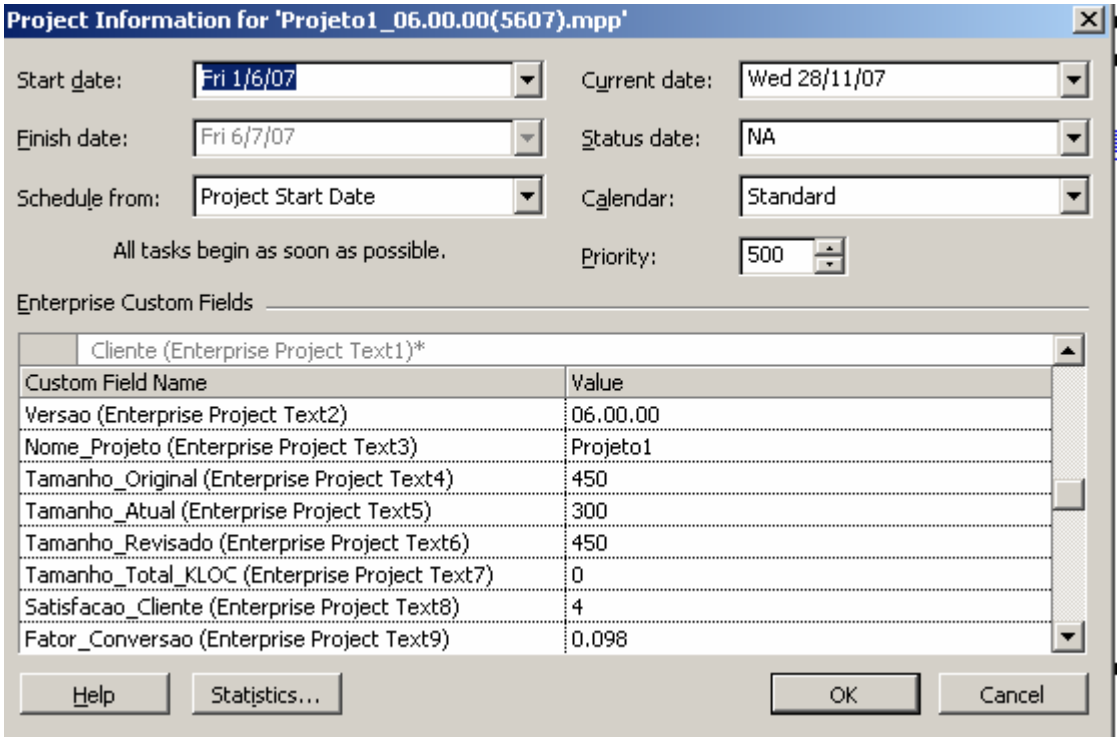
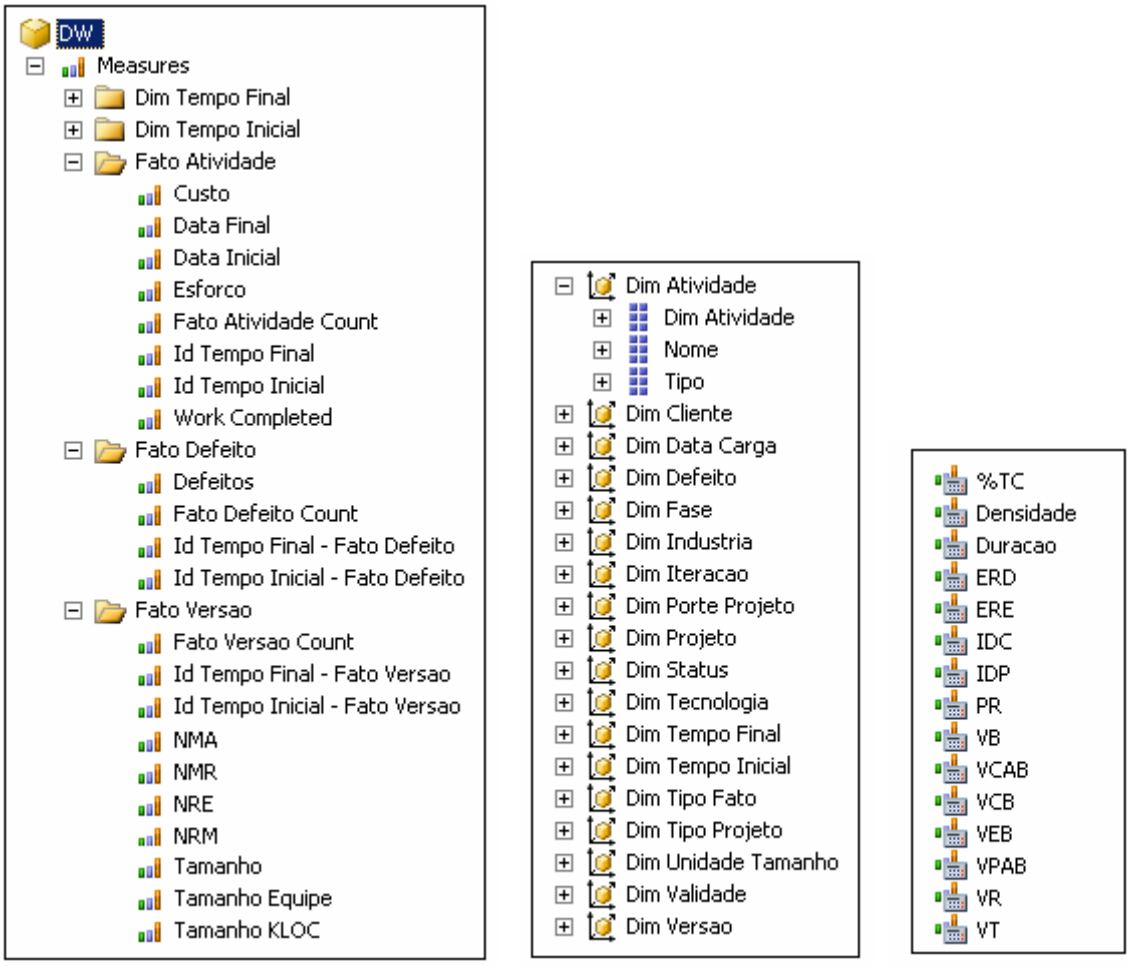


Figura 34 - Campos personalizados do *MS Project.*

7.4 Cubo *OLAP*

Para a realização dos testes foi criado um modelo *OLAP*, com o auxílio da ferramenta de *BI* do *SQL Server*. O mesmo é composto pelas tabelas fatos e as dimensões propostas no modelo analítico (Seção 6.3), juntamente com a inclusão da Dim_Validade, que determina se os valores do fato são válidos no presente ou não (Seção 6.3.3). Por padrão, essa ferramenta determina que as dimensões Dim_Tempo_Final e Dim_Tempo_Inicial, que armazenam estampas de tempo, também devam ser consideradas tabelas fato.

A Figura 35 ilustra: a) as tabelas fato e as suas métricas diretas aditivas e semi-aditivas, b) as dimensões do cubo, e c) as métricas derivadas. A Dim_Atividade foi expandida para mostrar seus atributos e exemplificar o modo de visualização proporcionado pela ferramenta. As métricas derivadas são calculadas por intermédio de quinze *scripts MDX*, executados durante a geração do cubo. A Figura 36 ilustra o *script* desenvolvido para realizar o cálculo da métrica derivada **VCB**, onde primeiramente é testada uma condição para verificar se o tipo do fato é Original. Caso a condição seja verdadeira, o valor é determinado conforme as equações presentes no *script*. Caso contrário, o *script* testa mais uma condição, onde verifica se o tipo do fato é Revisado, seguindo o mesmo procedimento anterior. Se o fato não atender nenhuma das duas condições, a métricas **VCB** recebe valor nulo (*null*).



(a) **(b)** **(c)**

Figura 35 – (a) Fatos, (b) Dimensões e (c) métricas derivadas.

```
CREATE MEMBER CURRENTCUBE.[MEASURES].[VCB]
 AS (IIF([Dim Tipo Fato].[Nome].currentmember.properties("Name")
= 'Original',
((([Dim Tipo Fato].[Nome].&[Real],[Measures].[Custo])
-[Measures].[Custo])*100)/
([Measures].[Custo]),
IIF([Dim Tipo Fato].[Nome].currentmember.properties("Name") =
'Revisado',
((([Dim Tipo Fato].[Nome].&[Real],[Measures].[Custo])-
([Measures].[Custo]))*100)/
([Measures].[Custo]),null))),
FORMAT_STRING = "Standard",
NON_EMPTY_BEHAVIOR = { [Custo] },
VISIBLE = 1;
```

Figura 36 – *Script MDX* para o cálculo da VCB.

7.5 Testes das Versões Encerradas

Para testar a mensuração da qualidade de *software* das versões encerradas (análise), foram realizadas consultas nas três tabelas fato principais: Fato_Atividade, Fato_Versão e Fato_Defeito. Os próximos itens apresentam os resultados obtidos em cada uma delas.

- Fato_Atividade:

A Figura 37 apresenta os dados da solução segundo as seguintes perspectivas de análise: versão, tipo do fato, fase, tipo da atividade e nome da atividade. Por motivos de legibilidade a dimensão iteração foi suprimida da consulta. Para cada uma dessas perspectivas também são apresentados totais que mostram os diferentes níveis de sumarização. No exemplo ilustrado, são disponibilizadas métricas das áreas de qualidade: Tempo, Esforço e Custo. Cabe salientar que as equações das métricas derivadas **VCB** e **VB** consideram o horário das datas (Data Inicial e Final), e que essas últimas apresentam o formato mês/dia/ano. É possível concluir que a solução apresenta as métricas conforme a hierarquia de cronograma apresentada na Figura 32, considerando as diferentes perspectivas de análise e os níveis de sumarização.

A Figura 38 apresenta métricas de Esforço e Custo para todas as versões encerradas, considerando os três tipos de fatos, reforçando o suporte a sumarizações em diferentes níveis. As métricas de **VCB** e **VB** são calculadas para fatos originais e revisados, em função dos valores reais. Dessa forma, os fatos reais não possuem valores para essas métricas. Percebe-se que a solução totaliza Esforço e Custo, para cada uma das versões. Além disso, é importante salientar que o **%TC**, considerado primordial à monitoração, apresenta um somatório correto.

Por ser tratar de um valor percentual, o **%TC** não pode ser obtido a partir do simples somatório entre os níveis, exigindo que o seu valor seja calculado em função do seu nível e do número de níveis inferiores. Por exemplo, uma fase formada por duas atividades A e B, onde cada uma represente 50% do esforço, se a atividade A apresentar **%TC** igual a 100 e B igual a 50, o valor correto é de **%TC** é 75.

Nome	Nome	Nome	Tipo	Nome	Esforco	Custo	%TC	Data Inicial	Data Final	VCB	VB
⊟ 01.00.00	⊟ Original	⊟ Design	⊟ Qualidade	DES - OS1	2.5	30	100,00	1/10/2007 8:00:00 AM	1/10/2007 5:00:00 PM	60,00	1.600,00
				DES - OS2	2.5	30	100,00	1/12/2007 8:00:00 AM	1/12/2007 5:00:00 PM	-20,00	1.333,33
				Total	5	60	100,00	1/10/2007 8:00:00 AM	1/12/2007 5:00:00 PM	20,00	210,53
			⊟ Trabalho	DES - OS1	20	240	100,00	1/2/2007 8:00:00 AM	1/4/2007 5:00:00 PM	5,00	0,00
				DES - OS2	19	228	100,00	1/4/2007 8:00:00 AM	1/9/2007 5:00:00 PM	5,26	-75,23
				Total	39	468	100,00	1/2/2007 8:00:00 AM	1/9/2007 5:00:00 PM	5,13	-54,83
			Total		44	528	100,00	1/2/2007 8:00:00 AM	1/12/2007 5:00:00 PM	6,82	48,19
		Total			44	528	100,00	1/2/2007 8:00:00 AM	1/12/2007 5:00:00 PM	6,82	48,19
	⊟ Real	⊟ Design	⊟ Qualidade	DES - OS1	4	48	100,00	1/16/2007 8:00:00 AM	1/16/2007 5:00:00 PM		
				DES - OS2	2	24	100,00	1/17/2007 8:00:00 AM	1/17/2007 5:00:00 PM		
				Total	6	72	100,00	1/16/2007 8:00:00 AM	1/17/2007 5:00:00 PM		
			⊟ Trabalho	DES - OS1	21	252	100,00	1/2/2007 8:00:00 AM	1/4/2007 5:00:00 PM		
				DES - OS2	20	240	100,00	1/4/2007 5:00:00 PM	1/5/2007 3:57:24 PM		
				Total	41	492	100,00	1/2/2007 8:00:00 AM	1/5/2007 3:57:24 PM		
			Total		47	564	100,00	1/2/2007 8:00:00 AM	1/17/2007 5:00:00 PM		
		Total			47	564	100,00	1/2/2007 8:00:00 AM	1/17/2007 5:00:00 PM		
	Total				91	1092	100,00	1/2/2007 8:00:00 AM	1/17/2007 5:00:00 PM		
Total geral					91	1092	100,00	1/2/2007 8:00:00 AM	1/17/2007 5:00:00 PM		

Figura 37 – Resultados da Fato_Atividade para uma versão.

Nome	Nome	Esforco	Custo	%TC	Data Inicial	Data Final	VCB	VB
⊟ 01.00.00	Original	283.1	3397.2	100,00	1/2/2007 8:00:00 AM	1/31/2007 5:00:00 PM	18,09	146,00
	Real	334.3	4011.6	100,00	1/2/2007 8:00:00 AM	3/15/2007 2:20:00 PM		
	Revisado	304	3648	100,00	1/2/2007 8:00:00 AM	1/31/2007 5:00:00 PM	9,97	146,00
	Total	921.4	11056.8	100,00	1/2/2007 8:00:00 AM	3/15/2007 2:20:00 PM		
⊟ 02.00.00	Original	222.2	2666.4	100,00	2/1/2007 8:00:00 AM	2/28/2007 5:00:00 PM	32,31	28,69
	Real	294	3528	100,00	2/1/2007 8:00:00 AM	3/8/2007 1:28:48 PM		
	Revisado	270.5	3246	100,00	2/1/2007 8:00:00 AM	2/28/2007 5:00:00 PM	8,69	28,69
	Total	786.7	9440.4	100,00	2/1/2007 8:00:00 AM	3/8/2007 1:28:48 PM		
⊟ 03.00.00	Original	248	2976	100,00	3/1/2007 8:00:00 AM	4/3/2007 5:00:00 PM	562,30	38,18
	Real	328.5	19710	100,00	3/1/2007 8:00:00 AM	4/16/2007 10:48:24 AM		
	Revisado	284	17040	100,00	3/1/2007 8:00:00 AM	4/3/2007 5:00:00 PM	15,67	38,18
	Total	860.5	39726	100,00	3/1/2007 8:00:00 AM	4/16/2007 10:48:24 AM		
⊟ 04.00.00	Original	152.5	9150	100,00	3/21/2007 8:00:00 AM	5/1/2007 5:00:00 PM	-65,44	-0,02
	Real	263.5	3162	100,00	4/2/2007 8:00:00 AM	5/1/2007 4:50:24 PM		
	Revisado	194	2328	100,00	3/21/2007 8:00:00 AM	5/1/2007 5:00:00 PM	35,82	-0,02
	Total	610	14640	100,00	3/21/2007 8:00:00 AM	5/1/2007 5:00:00 PM		
⊟ 05.00.00	Original	289	3468	100,00	5/2/2007 8:00:00 AM	6/12/2007 5:00:00 PM	31,14	-34,40
	Real	379	4548	100,00	5/2/2007 8:00:00 AM	5/29/2007 11:26:24 AM		
	Revisado	329.5	3954	100,00	5/2/2007 8:00:00 AM	6/12/2007 5:00:00 PM	15,02	-34,40
	Total	997.5	11970	100,00	5/2/2007 8:00:00 AM	6/12/2007 5:00:00 PM		
Total geral		4176.1	86833.2	100,00	1/2/2007 8:00:00 AM	6/12/2007 5:00:00 PM		

Figura 38 – Resultados da Fato_Atividade com mais de uma versão.

- Fato_Versão:

A Figura 39 ilustra os resultados obtidos a partir da consulta da tabela Fato_Versão e Fato_Atividade, onde são apresentados valores das áreas de qualidade Tamanho e Requisito. Esta solução relaciona métricas entre diferentes tabelas fato. Isso pode ser comprovado pela presença da métrica de custo, pertencente à tabela Fato_Atividade, na consulta da

Fato_Versão. Além disso, também é possível notar que a solução apresenta valores consistentes para esse cruzamento, já que os custos possuem os mesmos resultados ilustrados na Figura 39. Os valores mostrados na Figura 39 apresentam tanto métricas diretas (**NRA**, **NRE**, **NRM** e **NRO**), como derivadas (**VR**).

Nome	Nome	Custo	VCB	ERD	%TC	NRA	NRE	NRM	NRO	VR	Tamanho	VT
⊟ 01.00.00	Original	3397.2	18,09	5,00	100,00	19	0	6	24	1,04	470	8,23
	Real	4011.6		5,00	100,00	19	0	6	24	1,04	554	
	Revisado	3648	9,97	5,00	100,00	19	0	6	24	1,04	500	5,29
	Total	11056.8		5,00	100,00	19	0	6	24	1,04	1524	
⊟ 02.00.00	Original	2666.4	32,31	51,87	100,00	15	3	6	9	2,67	500	4,90
	Real	3528		51,87	100,00	15	3	6	9	2,67	550	
	Revisado	3246	8,69	51,87	100,00	15	3	6	9	2,67	522	2,74
	Total	9440.4		51,87	100,00	15	3	6	9	2,67	1572	
⊟ 03.00.00	Original	2976	562,30	58,67	100,00	3	0	4	2	3,50	550	4,90
	Real	19710		58,67	100,00	3	0	4	2	3,50	600	
	Revisado	17040	15,67	58,67	100,00	3	0	4	2	3,50	575	2,45
	Total	39726		58,67	100,00	3	0	4	2	3,50	1725	
⊟ 04.00.00	Original	9150	-65,44	61,99	100,00	7	0	4	3	3,67	453	7,74
	Real	3162		61,99	100,00	7	0	4	3	3,67	532	
	Revisado	2328	35,82	61,99	100,00	7	0	4	3	3,67	500	3,14
	Total	14640		61,99	100,00	7	0	4	3	3,67	1485	
⊟ 05.00.00	Original	3468	31,14	72,65	100,00	8	0	2	35	0,29	460	4,90
	Real	4548		72,65	100,00	8	0	2	35	0,29	510	
	Revisado	3954	15,02	72,65	100,00	8	0	2	35	0,29	485	2,45
	Total	11970		72,65	100,00	8	0	2	35	0,29	1455	
Total geral		86833.2		58,50	100,00	19	3	6	35	0,80	7761	

Figura 39 – Resultado da Fato_Versão e Fato_Atividade.

- Fato_Defeito:

A Figura 40 contém valores das métricas de defeitos (Defeitos e **ERD**), filtradas para todas as versões, da fase de Design, de um determinado projeto classificado como de grande porte. Assim, é possível concluir que a solução permite realizar consultas de defeitos segundos condições pré-definidas.

Nome	Nome	Nome	Nome	Defeitos	ERD
⊟ Projeto1	⊟ Grande	⊟ Design	01.00.00	3920	5,00
			02.00.00	5992	51,87
			03.00.00	17276	58,67
			04.00.00	14952	61,99
			05.00.00	13720	72,65
			06.00.00	3220	53,04
			Total	59080	58,20
		Total		59080	58,20
	Total			59080	58,20
Total geral				59080	58,20

Figura 40 – Resultado da Fato_Defeito.

7.6 Teste de Latência e Freqüência de Coleta

Para testar os aspectos de latência de dados e de freqüência da coleta dos mesmos, foram feitas as cargas dos cronogramas em andamento, de maneira seqüencial, minimizando o intervalo de tempo entre as mesmas. A Tabela 16 mostra o registro das datas e horas iniciais e finais, de cada passo realizado para a carga dos cronogramas, nos testes feitos em laboratório. Nela pode ser constatado que uma carga completa de um cronograma, com todo o processo de ETC automatizado, leva em torno de 4,6 minutos. Logo, pode-se concluir que o processo de ETC proposto atende a freqüência de coleta na ordem de horas e, por conseqüência, a latência de mesma ordem de grandeza temporal.

Tabela 16 - Tempos dos Passos de Carga.

Versão 06.00.00		Data Inicial	Data Final	Tempo
Carga 04/06/2007	**Passo 1**	5/12/2007	5/12/2007	00:00:25
	Passo 2	5/12/2007	5/12/2007	00:00:49
	Passo 3	5/12/2007	5/12/2007	00:00:29
	Passo 4	5/12/2007	5/12/2007	00:01:50
Total		**5/12/2007**	**5/12/2007**	**00:03:33**
Carga 04/06/2007	**Passo 1**	5/12/2007	5/12/2007	00:00:44
	Passo 2	5/12/2007	5/12/2007	00:00:53
	Passo 3	5/12/2007	5/12/2007	00:00:29
	Passo 4	5/12/2007	5/12/2007	00:04:23
Total		**5/12/2007**	**5/12/2007**	**00:06:29**
Carga 06/06/2007	**Passo 1**	5/12/2007	5/12/2007	00:01:11
	Passo 2	5/12/2007	5/12/2007	00:00:48
	Passo 3	5/12/2007	5/12/2007	00:00:55
	Passo 4	5/12/2007	5/12/2007	00:02:39
Total		**5/12/2007**	**5/12/2007**	**0:05:33**

7.7 Testes das Versões em Andamento

Para testar a mensuração da qualidade de *software* das versões em andamento (monitoração), segundo a técnica de *EVA*, foram realizadas consultas na tabela Fato_Atividade, em diferentes datas de carga. Os próximos itens apresentam os resultados obtidos em cada uma delas.

- Primeira Carga - 04/06/2007

A Figura 41 apresenta os valores da primeira carga com os dados do cronograma da versão 06.00.00, onde são mostrados somente valores originais. No momento da carga, a versão ainda não tinha entrado em execução, tendo valor zero em todos os valores de **%TC** e de esforço das suas atividades. Por motivos de legibilidade, as dimensões Validade e Versão foram suprimidas dessa figura, bem como os valores totais da versão em andamento.

Nome	Nome	Tipo	Nome	Esforco	Custo	%TC	Data Inicial	Data Final
⊟ Original	⊟ Design	⊟ Trabalho	DES - OS1	17	204	0,00	6/1/2007 8:00:00 AM	6/4/2007 5:00:00 PM
			DES - OS2	14	168	0,00	6/1/2007 8:00:00 AM	6/4/2007 5:00:00 PM
			DES - OS3	21	252	0,00	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM
			Total	52	624	0,00	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM
		⊟ Revisão	DES - OS1	4	48	0,00	6/4/2007 8:00:00 AM	6/4/2007 5:00:00 PM
			DES - OS2	7	84	0,00	6/4/2007 8:00:00 AM	6/5/2007 5:00:00 PM
			DES - OS3	5	60	0,00	6/5/2007 8:00:00 AM	6/5/2007 5:00:00 PM
			Total	16	192	0,00	6/4/2007 8:00:00 AM	6/5/2007 5:00:00 PM
		Total		68	816	0,00	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM
	Total			68	816	0,00	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM
⊟ Real	⊟ Design	⊟ Trabalho	DES - OS1	0	0	0,00	6/1/2007 8:00:00 AM	6/4/2007 5:00:00 PM
			DES - OS2	0	0	0,00	6/5/2007 8:00:00 AM	6/6/2007 5:00:00 PM
			DES - OS3	0	0	0,00	6/7/2007 8:00:00 AM	6/11/2007 5:00:00 PM
			Total	0	0	0,00	6/1/2007 8:00:00 AM	6/11/2007 5:00:00 PM
		⊟ Revisão	DES - OS1	0	0	0,00	6/12/2007 8:00:00 AM	6/13/2007 5:00:00 PM
			DES - OS2	0	0	0,00	6/14/2007 8:00:00 AM	6/14/2007 5:00:00 PM
			DES - OS3	0	0	0,00	6/15/2007 8:00:00 AM	6/15/2007 5:00:00 PM
			Total	0	0	0,00	6/12/2007 8:00:00 AM	6/15/2007 5:00:00 PM
		Total		0	0	0,00	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM
	Total			0	0	0,00	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM
Total				68	816	0,00	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM
				68	816	0,00	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM
				68	816	0,00	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM

Figura 41 – Resultados da Primeira Carga.

- Segunda Carga - 05/06/2007

A Figura 42 lista os resultados da segunda carga, em dois blocos: o superior refere-se a fatos do *baseline* original; o inferior, a fatos reais. As atividades da fase de Design do tipo Trabalho (bloco superior) foram iniciadas, apresentando 44,30% para **%TC**. A origem desses valores está na Figura 33. Para cada uma dessas atividades a solução calcula as métricas de *EVA* (Tabela 10). Como as atividades do Tipo Revisão ainda não foram iniciadas, seus valores para monitoração não foram calculados. Por sua vez os fatos do tipo Real (bloco inferior) também não apresentam valores, mas são utilizados para o cálculo das métricas *EVA*. Além de computar os valores de *EVA* para as tarefas de maneira individualmente, a solução também propaga esse comportamento para os demais níveis de hierarquia. Isso pode ser percebido pelo Total dessas métricas, apresentados para Tipo de Atividade (Trabalho), Fase (Design) e Tipo do Fato (Original).

Os indicadores de qualidade, para as métricas **VB** e **VCB,** utilizados pela operação de *software* (Capítulo 5), são interpretados conforme a Tabela 17. Esses indicadores, apesar de destinarem-se apenas à análise, têm sido usados para monitoração. A partir da avaliação dos valores dessas métricas mostrados na Figura 42, segundo esses indicadores, constata-se que as atividades estão sendo realizadas dentro do prazo mas acima do custo esperado. Por exemplo, para a atividade DES-OS2, **VCB** = -42,86% e o **IDC** = 0,88. Logo, pela técnica de *EVA* a atividade está acima do custo, apesar do indicador de qualidade acusar Aceitável para **VCB**. Isso mostra a conveniência do uso de métricas *EVA* para monitoração.

Nome	Tipo	Nome	Esforco	Custo	%TC	Data Inicial	Data Final	VB	VPAB	IDP	VCB	VCAB	IDC
⊟ Design	⊟ Trabalho	DES - OS1	17	204	50,00	6/1/2007 8:00:00 AM	6/4/2007 5:00:00 PM	0,00	-167,48	0,38	-47,06	-6,00	0,94
		DES - OS2	14	168	50,00	6/1/2007 8:00:00 AM	6/4/2007 5:00:00 PM	59,26	-137,93	0,38	-42,86	-12,00	0,88
		DES - OS3	21	252	33,00	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM	131,40	-173,64	0,32	-61,90	-12,84	0,87
		Total	52	624	44,33	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM	131,40	-359,25	0,44	-51,92	-23,36	0,92
	⊟ Revisão	DES - OS1	4	48	0,00	6/4/2007 8:00:00 AM	6/4/2007 5:00:00 PM	2.400,00			-100,00		
		DES - OS2	7	84	0,00	6/4/2007 8:00:00 AM	6/5/2007 5:00:00 PM	654,55			-100,00		
		DES - OS3	5	60	0,00	6/5/2007 8:00:00 AM	6/5/2007 5:00:00 PM	2.666,67			-100,00		
		Total	16	192	0,00	6/4/2007 8:00:00 AM	6/5/2007 5:00:00 PM	727,27			-100,00		
	Total		68	816	22,17	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM	228,57	-650,66	0,22	-63,24	-119,12	0,60
Total			68	816	22,17	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM	228,57	-650,66	0,22	-63,24	-119,12	0,60
⊟ Design	⊟ Trabalho	DES - OS1	9	108	50,00	6/1/2007 8:00:00 AM	6/4/2007 5:00:00 PM						
		DES - OS2	8	96	50,00	6/5/2007 8:00:00 AM	6/6/2007 5:00:00 PM						
		DES - OS3	8	96	33,00	6/5/2007 8:00:00 AM	6/11/2007 10:58:12 AM						
		Total	25	300	44,33	6/1/2007 8:00:00 AM	6/11/2007 10:58:12 AM						
	⊟ Revisão	DES - OS1	0	0	0,00	6/12/2007 8:00:00 AM	6/13/2007 5:00:00 PM						
		DES - OS2	0	0	0,00	6/14/2007 8:00:00 AM	6/14/2007 5:00:00 PM						
		DES - OS3	0	0	0,00	6/15/2007 8:00:00 AM	6/15/2007 5:00:00 PM						
		Total	0	0	0,00	6/12/2007 8:00:00 AM	6/15/2007 5:00:00 PM						
	Total		25	300	22,17	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM						
Total			25	300	22,17	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM						

Figura 42 – Resultados da Segunda Carga.

Tabela 17 - Indicadores de Qualidade, segundo referências de mercado [HPC04].

Métricas Derivadas	Aceitável	Alerta	Inaceitável
VB e **VCB**	<=5%	>5% e <=10%	>10%

- Terceira Carga – 06/06/2007

A Figura 43 ilustra os valores coletados no terceiro dia de carga. A partir dos mesmos é possível perceber que as atividades de Trabalho da fase de Design, juntamente com a atividade DES-01 do tipo Revisão, já foram concluídas, e por essa razão não possuem métricas de monitoração. Porém, a atividade de Revisão DES-02 encontra-se em andamento e a DES-03 ainda não começou. A solução apresenta valores de monitoração totais para a fase de Design, para valores do *baseline* original e real, e para a versão 06.00.00 (última linha da figura).

Nome	Tipo	Nome	Custo	Esforco	%TC	Data Inicial	Data Final	VB	VPAB	IDP	VCB	VCAB	IDC
⊟ Design	⊟ Trabalho	DES - OS1	204	17	100,00	6/1/2007 8:00:00 AM	6/4/2007 5:00:00 PM	0,00			0,00		
		DES - OS2	168	14	100,00	6/1/2007 8:00:00 AM	6/4/2007 5:00:00 PM	59,26			7,14		
		DES - OS3	252	21	100,00	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM	131,40			52,62		
		Total	624	52	100,00	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM	131,40			23,17		
	⊟ Revisão	DES - OS1	48	4	100,00	6/4/2007 8:00:00 AM	6/4/2007 5:00:00 PM	2.332,96			0,00		
		DES - OS2	84	7	47,00	6/4/2007 8:00:00 AM	6/5/2007 5:00:00 PM	490,81	-110,70	0,26	-78,57	21,48	2,19
		DES - OS3	60	5	0,00	6/5/2007 8:00:00 AM	6/5/2007 5:00:00 PM	2.666,67			-100,00		
		Total	192	16	49,00	6/4/2007 8:00:00 AM	6/5/2007 5:00:00 PM	727,27	-249,19	0,27	-65,63	28,08	1,43
	Total		816	68	74,50	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM	228,57	-410,14	0,60	2,28	-226,68	0,73
Total			816	68	74,50	6/1/2007 8:00:00 AM	6/5/2007 5:00:00 PM	228,57	-410,14	0,60	2,28	-226,68	0,73
⊟ Design	⊟ Trabalho	DES - OS1	204	17	100,00	6/1/2007 8:00:00 AM	6/4/2007 5:00:00 PM						
		DES - OS2	180	15	100,00	6/5/2007 8:00:00 AM	6/6/2007 5:00:00 PM						
		DES - OS3	384.6	32.05	100,00	6/5/2007 8:00:00 AM	6/11/2007 10:58:00 AM						
		Total	768.6	64.05	100,00	6/1/2007 8:00:00 AM	6/11/2007 10:58:00 AM						
	⊟ Revisão	DES - OS1	48	4	100,00	6/11/2007 10:58:00 AM	6/13/2007 10:58:00 AM						
		DES - OS2	18	1.5	47,00	6/11/2007 10:58:00 AM	6/12/2007 10:58:00 AM						
		DES - OS3	0	0	0,00	6/15/2007 8:00:00 AM	6/15/2007 5:00:00 PM						
		Total	66	5.5	49,00	6/11/2007 10:58:00 AM	6/15/2007 5:00:00 PM						
	Total		834.6	69.55	74,50	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM						
Total			834.6	69.55	74,50	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM						
			1650.6	137.55	74,50	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM						
			1650.6	137.55	74,50	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM						
			1650.6	137.55	74,50	6/1/2007 8:00:00 AM	6/15/2007 5:00:00 PM						

Figura 43 – Resultados da Terceira Carga.

Com base nos valores apresentados pelas Figuras 41, 42 e 43 é possível verificar os resultados obtidos com a execução da carga incremental, segundo as três regras propostas na Seção 6.5.4. A condição da Regra 1 foi verificada como verdadeira, somente, durante a carga do dia 03/06/2007, pois todos os valores do cronograma estavam sendo inseridos pela primeira vez no *DW*. Já na carga do dia 04/06/2007, apenas as condições das Regras 2 e 3 foram consideradas verdadeiras, devido à presença de novos valores para todas as atividades de trabalho, da fase de Design, já inseridas previamente no *DW*. Dessa forma, o período de validade dos fatos anteriormente armazenados tiveram que ser encerrados, tomando com data final a data de carga, como pode ser visto na consulta feita na tabela Fato_Atividade, selecionando apenas a Atividade DES-OS1, do tipo Trabalho e da fase de Design, mostrada na Figura 44.

Results | Messages

	Validade	Tipo_Fato	Fase	Atividade	Tipo_Atividade	Esforco	%TC	Data_Validade_Inicial	Data_Validade_Final
1	Válido	Original	Design	DES-OS1	Trabalho	17	100	2007-06-06 19:00:00.000	NULL
2	Válido	Real	Design	DES-OS1	Trabalho	0	100	2007-06-06 19:00:00.000	NULL
3	Inválido	Original	Design	DES-OS1	Trabalho	17	0	2007-06-04 19:00:00.000	2007-06-05 18:59:00.000
4	Inválido	Original	Design	DES-OS1	Trabalho	17	50	2007-06-05 19:00:00.000	2007-06-06 18:59:00.000
5	Inválido	Real	Design	DES-OS1	Trabalho	0	0	2007-06-04 19:00:00.000	2007-06-05 18:59:00.000
6	Inválido	Real	Design	DES-OS1	Trabalho	0	50	2007-06-05 19:00:00.000	2007-06-06 18:59:00.000

Figura 44 – Validades da Atividade DES-OS1.

7.8 Considerações sobre os Testes

A partir dos resultados obtidos, por intermédio da exploração do cubo *OLAP* gerado para testar a solução, foi possível concluir que: 1) o modelo analítico proposto oferece suporte adequado à análise e à monitoração da mensuração da qualidade de *software*, permitindo diferentes perspectivas de análise e níveis de sumarização; 2) a solução apresenta de forma consistente as métricas de *EVA*, incorporadas no Programa de Métricas; 3) o processo de ETC das métricas a serem armazenadas no Repositório de Dados é automatizado, possibilitando cargas freqüentes, com latência na ordem de horas e baixa intrusão, desde que seja considerada a sua automação, conforme os componentes apresentados na solução.

É importante mencionar, que a construção do cubo *OLAP*, descrito na Seção 7.4, foi fundamental para a geração dos resultados e exploração das métricas. A utilização do mesmo permitiu concluir que a proposta do *SPDW+* atende os pré-requisitos da solução original [BEC06], de uma maneira mais elegante e abrangente.

8 CONSIDERAÇÕES FINAIS

Este trabalho propõe um processo automatizado de ETC de métricas da qualidade de *software*, a serem armazenados em um Repositório de Dados. Para tanto, define um instrumento de referência, ilustrado pela Tabela 1, que apresenta a mensuração da qualidade a partir de três dimensões ortogonais: 1) nível de mensuração, 2) atividades de supervisão, e 3) plataforma computacional. A partir dessas dimensões, é realizada uma discussão sobre o cenário de desenvolvimento de *software*, caracterizando os principais aspectos que envolvem a definição de uma arquitetura computacional para automação, parcial ou total, desse cenário.

Com base nessas dimensões e nos aspectos definidos pelo instrumento, foi efetuada uma pesquisa bibliográfica para encontrar trabalhos relacionados com este tema de pesquisa. O resultado da mesma é apresentado no Capítulo 4, a partir do qual pôde-se perceber que os trabalhos relatados atendem parcialmente os aspectos definidos no instrumento de referência.

Para caracterizar um cenário real de aplicação desta pesquisa, foi realizado um estudo de caso, relatado no Capítulo 5, onde é apresentado o ambiente de desenvolvimento de *software* de uma organização de TI, considerada de grande porte na América Latina e certificada *CMM3*. Nesse capítulo foi descrito o seu processo de ETC, destacando as suas facilidades e dificuldades.

A solução proposta denomina-se *SPDW+* e consiste em um ambiente de *data warehousing*, que oferece suporte à análise e à monitoração da mensuração da qualidade de *software*. A mesma contempla um modelo analítico elegante e abrangente, que permite a monitoração segundo a técnica *EVA*, a partir no nível de atividade. Além disso, o *SPDW+* define um processo automatizado de ETC das métricas que trata a carga incremental, a alta freqüência e a baixa latência e intrusão da coleta. Para isso, a solução é consistente com a arquitetura orientada a serviços. Foram especificados os seus principais componentes: (i) Rotina de Extração, (ii) *wrappers*, (iii) Rotinas de Limpeza e Transformação, (iv) Metadados de Projeto, (v) *DSA*, (vi) serviço de carga incremental e (vii) Repositório de Dados (*DW*).

Para avaliar esta solução foi efetuado um conjunto de testes, descrito no Capítulo 7, seguindo um número pré-definido de passos e a partir da exploração do conteúdo das métricas por intermédio de tabelas pivotantes. Dessa maneira, foi possível verificar: 1) a adequação do modelo analítico para análise e monitoração, desde o nível de atividades, 2) a efetividade das métricas *EVA* na monitoração de métricas, e 3) a eficácia do processo automatizado de ETC.

Essa solução é um importante avanço em relação a trabalhos anteriores [CUN05] por apresentar um modelo analítico elegante e abrangente, e especificar um ambiente com suporte à monitoração de métricas de *software*.

8.1 Trabalho Futuros

A continuidade da pesquisa vai, justamente, em propor uma solução estritamente consistente com o padrão arquitetural orientado a serviços para o ambiente de *data warehousing* proposto em [BEC06]. Em especial, pretende-se:

1. adicionar um módulo para previsão de métricas, baseado em técnicas de mineração [FIG07], na arquitetura do *SPDW+*.
2. permitir a evolução do Programa de Métricas e dos processos organizacionais (*OSSP* e *PDP*), a partir de esquemas, conforme o trabalho proposto por [BOG06].
3. investigar outros modelos de desenvolvimento de *software* para verificar a adequação desta solução.
4. pesquisar quais os aspectos que merecem ser melhorados para que a arquitetura do *SPDW+* possa considerada de fato uma AOS.

REFERÊNCIAS

[AAL03] AALST, W.; DUMAS, M.; HOFSTEDE, A. Web service composition languages: Old wine in new bottles?. In: EUROMICRO Conference "New Waves in System Architecture" (EUROMICRO'03), 29th, 2003, Belek-Antalya, Turkey. **Proceedings…** Los Alamitos: IEEE Computer Society Press, 2003, p. 298-307.

[ABN05] ABNT - Associação Brasileira de Normas Técnicas. **NBR ISO 9000 – Gestão de Qualidade**. Rio de Janeiro: ABNT/CB-25 QUALIDADE, 2005. 35 p.

[ALO04] ALONSO, G.; CASATI, F.; KUNO, H.; MACHIRAJU, V. **Web Services – Concepts, Architectures and Applications**. Berlin: Springer, 2004. 354 p.

[BEC06] BECKER, K.; RUIZ, D.; NOVELLO, T.; CUNHA, V. SPDW: a Software Development Process Performance Data Warehousing Environment. In: Software Engineering Workshop (SEW'06), 30, 2006, Bethesda, MD. **Proceedings…** Los Alamitos: IEEE Computer Society Press, 2006, p. 107-118.

[BOG06] BOGONI, L. **Um Método Evolutivo para Aplicação de Programas de Métricas em Processos de Desenvolvimento de Software**. 2006. 79 f. Dissertação (Mestrado em Ciência da Computação) – Faculdade de Informática, PUCRS, Porto Alegre, 2006.

[BOR07] BORLAND SOFTWARE CORPORATION. **CaliberRM.** Disponível em: http://www.borland.com/br/products/caliber/index.html. Acesso em: 1 nov. 2007.

[CAS02a] CASATI, F. A Conversation on Web Services: what´s new, what´s true, what´s not. And what´s not. In: Workshop on Knowledge Transformation for the Semantic for the Semantic Web at the European Conference on Artificial Intelligence (KTSW-2002), 15, 2002, Lyon, France. **Electronic Proceedings…** Disponível em: http://www.cs.vu.nl/~borys/events/ktsw2002.pdf. Acesso em: 13 mar. 2007.

[CAS02b] CASATI, F.; DAYAL, U.; SAYAL, M.; SHAN, M. **Business Process Intelligence**. Hewlett Packard Technical Report HPL-2002119, 2004. Disponível em: http://www.hpl.hp.com/techreports/2002/HPL-2002-119.pdf. Acesso em: 06 jul. 2006.

[CAS05] CASTELLANOS, M.; CASATI, F.; DAYAL, U.; SHAN, M. iBOM: A Platform for Intelligent Business Operation Management. In: International Conference on Data Engineering (ICDE), 21, 2005, Tokyo. **Proceedings…** Los Alamitos: IEEE Computer Society Press, 2005, p. 1084-1095.

[CUN05] CUNHA, V. **Uma Abordagem Orientada a Serviços para Captura de Métricas de Processo de Desenvolvimento de Software**. 2005. 117 f. Dissertação (Mestrado em Ciência da Computação) – Faculdade de Informática, PUCRS, Porto Alegre, 2005.

[DEP07] DEPARTMENT OF DEFENSE. **Earned Value Management Implementation Guide**. Washington: United States of America Department of Defense, 1997. Disponível em: http://guidebook.dcma.mil/79/guidebook_process.htm. Acesso em: 15 jul. 2007.

[EMA97] EMAN, K.; MELO, W.; DROUIN, J. **SPICE: The Theory and Practice of Software Process Improvement and Capability Determination.** , v. 1. Los Alamitos, CA: IEEE Computer Society Press, 1997. 450 p.

[EMI06] EMIG, C.; WEISSER, J.; ABECK, S. Development of SOA-Based Software Systems: an Evolutionary Programming Approach. In: Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW), Guadeloupe, French Caribbean. **Proceedings…** Los Alamitos: IEEE Computer Society Press, 2006, p.182-188.

[FAC02] FACHIN, O. **Fundamentos de Metodologia**, 3ª Edição, São Paulo: Atlas, 2002. 200 p.

[FIG07] FIGUEIRA, F. **Aplicação de Métodos Preditivos em Programas de Métricas de Software**. 2007. Dissertação (Mestrado em Ciência da Computação) – Faculdade de Informática, PUCRS, Porto Alegre, 2007. (dissertação em finalização).

[FLE00] FLEMING, Q.; KOPPELMAN, J. **Earned Value Project Management**. 2nd edition, Pennsylvania, USA: Project Management Institute, 2000. 224 p.

[GOL04] GOLFARELLI, M.; RIZZI, S.; CELLA, I. Beyond data warehousing: what's next in business intelligence? In: ACM International Workshop on Data Warehousing and OLAP, 7, Washington, DC, USA, 2004. **Proceedings...** New York: ACM Press, 2004, p. 1-6.

[GRI04] GRIGORI, D.; CASATI, F.; CASTELLANOS, M.; DAYAL, U.; SAYAL, M. Business Process Intelligence. **Computers in Industry**, Amerterdan: The Netherlands, v.53, n.3, p. 321-343, 2004.

[HAN01] HAN, J.; KAMBER, M. **Data Mining:** Concepts and Techniques. San Francisco: Morgan Kaufmann, 2001. 550 p.

[HAY03] HAYNES, S. Institutional Metrics for the United States Marine Corps. In: Hawaii International Conference on System Sciences (HICSS), 36, Big Island, HI, 2003. **Proceedings…** Los Alamitos: IEEE Computer Society Press, 2003. p.231-240.

[HPC04] HP - Hewlett-Packard Company Brasil Ltda. **Programa de Métricas 2.3**. Porto Alegre: HP EAS Brasil, 2004. 13 p. (Relatório Técnico)

[HPC05a] HP - Hewlett-Packard Company Brasil Ltda. **Checklist de Carga 1.0**. Porto Alegre: HP EAS Brasil, 2005. 15 p. (Relatório Técnico)

[HPC05b] HP - Hewlett-Packard Company Brasil Ltda. **Guia de Captura de Defeitos 1.1**. Porto Alegre: HP EAS Brasil, 2005. 6 p. (Relatório Técnico)

[HPC05c] HP - Hewlett-Packard Company Brasil Ltda. **Guia de Registro de Esforço 1.0**. Porto Alegre: HP EAS Brasil, 2005. 6 p. (Relatório Técnico)

[HPC05d] HP - Hewlett-Packard Company Brasil Ltda. **Guia de Registro de Requisitos 1.0**. Porto Alegre: HP EAS Brasil, 2005. 6 p. (Relatório Técnico)

[HPC06a] HP - Hewlett-Packard Company Brasil Ltda. **M1 – Um Diagnóstico da Base Organizacional HP EAS Brasil 1.0**. Porto Alegre: HP EAS Brasil, 2006. 23 p. (Relatório Técnico)

[HPC06b] HP - Hewlett-Packard Company Brasil Ltda. **M3 - Sugestão de Métricas de Monitoração e Previsão HP EAS Brasil 1.1** Porto Alegre: HP EAS Brasil, 2006. 21 p. (Relatório Técnico)

[HPC06c] HP - Hewlett-Packard Company Brasil Ltda. **M11 – Modelo e Esquema da Base Organizacional para a Monitoração de Projetos na HP EAS Brasil 1.0.** Porto Alegre: HP EAS Brasil, 2006. 16 p. (Relatório Técnico)

[HPC07a] HP - Hewlett-Packard Company Brasil Ltda. **ME3 – Migração dos DTS Defeitos e Temporária para SSIS 1.0.** Porto Alegre: HP EAS Brasil, 2007. 25 p. (Relatório Técnico)

[HPC07b] HP - Hewlett-Packard Company Brasil Ltda. **M19 – Estudo sobre o Impacto do Modelo Analítico Estendido no Processo de Carga da BO HP EAS Brasil 1.0.** Porto Alegre: HP EAS Brasil, 2007. 31 p. (Relatório Técnico)

[HPC07c] HP - HEWLETT-PACKARD COMPANY. **HP Global Method for Application Services (HPGM-AS) – IQMS.** Disponível em: http://gm-as.hp.com/iqs/website/webpages/html_gd/index.asp. Acesso em: 11 nov. 2007.

[IBM07a] IBM. **IBM Rational ClearQuest**. Disponível em: http://www-306.ibm.com/software/awdtools/clearquest/. Acesso em: 11 nov. 2007.

[IBM07b] IBM. **IBM Rational RequisitePro**. Disponível em: http://www-111.ibm.com/ecatalog/Detail.*WS*s?locale=pt_BR&synkey=U107428M40511T21 . Acesso em: 10 nov. 2007.

[IEE98] ANSI/IEEE Std 1061-1998. **IEEE Standard for a Software Quality Metrics Methodology**, Piscataway, NJ: IEEE Standards Dept., 1998. 26 p.

[INM05] INMON, W. **Building the Data Warehouse**. Indianapolis, IN: John Wiley & Sons, Inc, 2005. 543 p.

[KAN03] KAN, S. **Metrics and Models in Software Quality Engineering**. Boston: Addison-Wesley, 2003. 528 p.

[KAN04] KANER, C.; BOND, W. Software Engineering Metrics: What do they measure and how do we know? In: International Software. Metrics Symposium (Metrics 2004), 10, 2004, Chicago, IL, USA. **Procedings…** Los Alamitos: IEEE Computer Society Press, 2004. p 1-12.

[KEE04] KEEN, M. et al. **Patterns: Implementing an SOA Using an Enterprise Service Bus**. IBM Redbooks. Disponível em: http://public-.boulder.ibm.com/Redbooks.nfs/0/2b0de69bb65f8a81285256e620078c4c9?OpenDocument. Acesso em: 12 Jul. 2004.

[KHO01] KHOSHGOFTAAR, M.; ALLEN, E.; JONES, W.; HUDEPOHL, J. Data Mining of Software Development Databases. **Software Quality Journal**, New York, NY, v. 9, n. 3, p. 161-176, nov. 2001.

[KIM98] KIMBALL, R. **Data Warehouse Toolkit**. New York, NY: John Wiley & Sons, Inc, 1998. 771 p.

[KOR07] KORZH.COM. **Clarity**. Capturado em: http://devtools.korzh.com/clarity/. Acesso em: 03 nov. 2007.

[KRO05] KROGDAHL, P.; LUEF, G.; STEINDL, C. Service-oriented agility: an initial analysis for the use of agile methods for SOA development. In: IEEE International Conference on Services Computing (SCC'05), 2005, Orlando, Florida, USA. **Proceedings…** Los Alamitos: IEEE Computer Society Press, 2005. p. 93-100.

[LEY03] LEYMAN, F. Web Services: Distributed Applications Without Limits. In: Database Systems For Business, Technology and Web BTW, 2003, Leipzig, Germany. **Proceedings…** Bonn Alemanha: Gesellschaft für Informatik (Lecture Notes in Informatics, V26), 2003. p. 2-23.

[MAN07] MANTIS. **Mantis Bug Tracker**. Disponível em: http://www.mantisbt.org/. Acesso em: 22 nov. 2007.

[MER07] MERCURY. **PPM – Project and Portfolio Management.** Disponível em: http://support.openview.hp.com/pdf/dst/ppm_Daylight_Savings_time_Update.pdf. Acessado em: 22 jan. 2007

[MIC07] MICROSOFT OFFICE ON-LINE. **Microsoft Office Project 2007**. Disponível em: http://office.microsoft.com/pt-br/project/FX100487771046.aspx. Acesso em: 21 nov. 2007.

[MSD07] MSDN SQL Server Developer Center. **Deprecated Features in SQL Server 2005 Integration Services**. Disponível em: http://msdn2.microsoft.com/en-us/library/ms403408.aspx. Acesso em:10 nov. 2007.

[NAV93] NAVATHE, S.; AHMED, R. Temporal Extensions to the Relational Model and SQL. In: NAVATHE, S., AHMED, R. **Temporal Databases: Theory, Design and Implementation**. Bridge Parkway: Benjamin/Cummings, 1993. p. 92-109.

[NEW02] NEWCOMER, E. . **Understanding Web Services XML, WSDL, SOAP and UDDI.** Boston: Addison-Wesley, 2002. 332 p.

[OAS06] OASIS. **Organization for the Advancement of Structured Information Standards**. Disponível em: http://www.oasis-open.org/home/index.php. Acesso em: 4 out. 2006.

[ORE06] O' REILLY JAVA.COM. **Java and Web Services Primer**. Disponível em: http://www.onjava.com/pub/a/onjava/2001/08/07/webservices.html. Acesso em 14 set. 2006.

[OTL01] OTLEY, D. Extending the Boundaries of Management Accounting Research: Developing Systems for Performance Measurement, **British Accounting Review**, v. 33, p. 243-261, 2001.

[PAL03] PALZA, E.; FUHRMAN, C.; ABRAN, A. Establishing a Generic and Multidimensional Measurement Repository in CMMI context. In: Annual IEEE/NASA Software Engineering Workshop, 28, 2003, Greenbelt, MD, USA. **Proceedings...** Los Alamitos: IEEE Computer Society Press, 2003, p.12-20.

[PMI04] Project Management Institute. **A Guide to the Project Management Body of Knowledge** (PMBOK Guide). 3rd Edition. Newton Square: Project Management Institute, 2004. 380 p.

[PRE04] PRESSMAN, R. **Software Engineering.** New York: McGraw-Hill, 2004. 888 p.

[RUI05] RUIZ, D.; BECKER, K.; NOVELLO, T.; CUNHA, V. A data warehousing environment to monitor metrics in software development processes. In: International Workshop on Business Process Monitoring & Performance Management (BPMPM 2005) on 16th Workshop on Database and Expert System Applications (DEXA05), 2005, Copenhagen. **Proceedings…** Los Alamitos: IEEE Computer Society Press, 2005. p. 936-940.

[SEI06] SEI - Software Engineering Institute. **CMMI for Development, Version 1.2**. Pittsburgh: Carnegie Mellon University and Software Engineering Institute, 2006. Disponível em: http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06tr008.pdf. Acesso em: 15 jan. 2007.

[SID02] SIDDIQUI, B. **Deploying Web Services with WSDL, Part 2: Simple Access Protocol (SOAP).** Disponível em: http://www-128.ibm.com/developerworks/library/*WS*-int*WSDL*2/. Acesso em: 5 out. 2006.

[SOF07] SOFTEX. **MPS.Br Capacitação e Empreendedorismo**. Disponível em: http://www.softex.br/mpsbr/_home/default.asp. Acesso em: 20 nov. 2007.

[SOM04] SOMMERVILLE, I. **Software Engineering**. 5th Edition. Boston: Addison-Wesley, 2004. 592 p.

[SQL07] **SQL Server 2000 – Data Transformation Services.** Disponível em: http://technet.microsoft.com/en-us/sqlserver/bb331744.aspx. Acesso em: 25 nov. 2007.

[THE07] THE MOZILLA CORPORATION. **Bugzilla**. Disponível em: http://www.bugzilla.org/. Acesso em: 21 nov. 2007.

[VAR04] VARGAS, R. **Microsoft Office Project 2003: Standard, Professional & Server**. Rio de Janeiro: Brasport, 2004. 549 p.

[W3C06a] W3C. **Extensible Markup Language**. Disponível em: http://www.w3.org/XML/. Acesso em: 3 out. 2006.

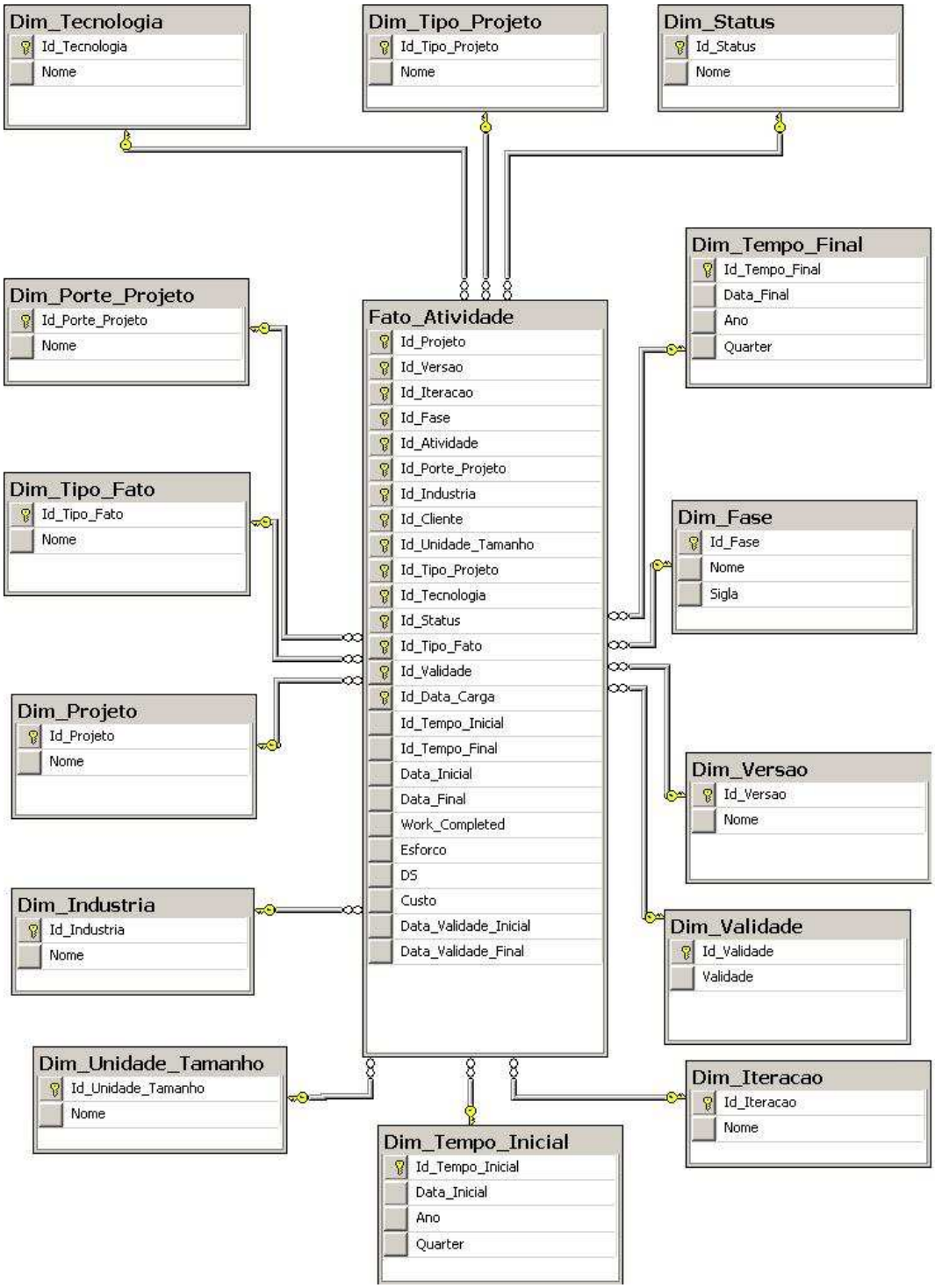
[W3C06b] W3C. **Simple Object Access Protocol**. Disponível em: http://www.w3.org/TR/2000/NOTE-*SOAP*-20000508/. Acesso em: 12 set. 2006.

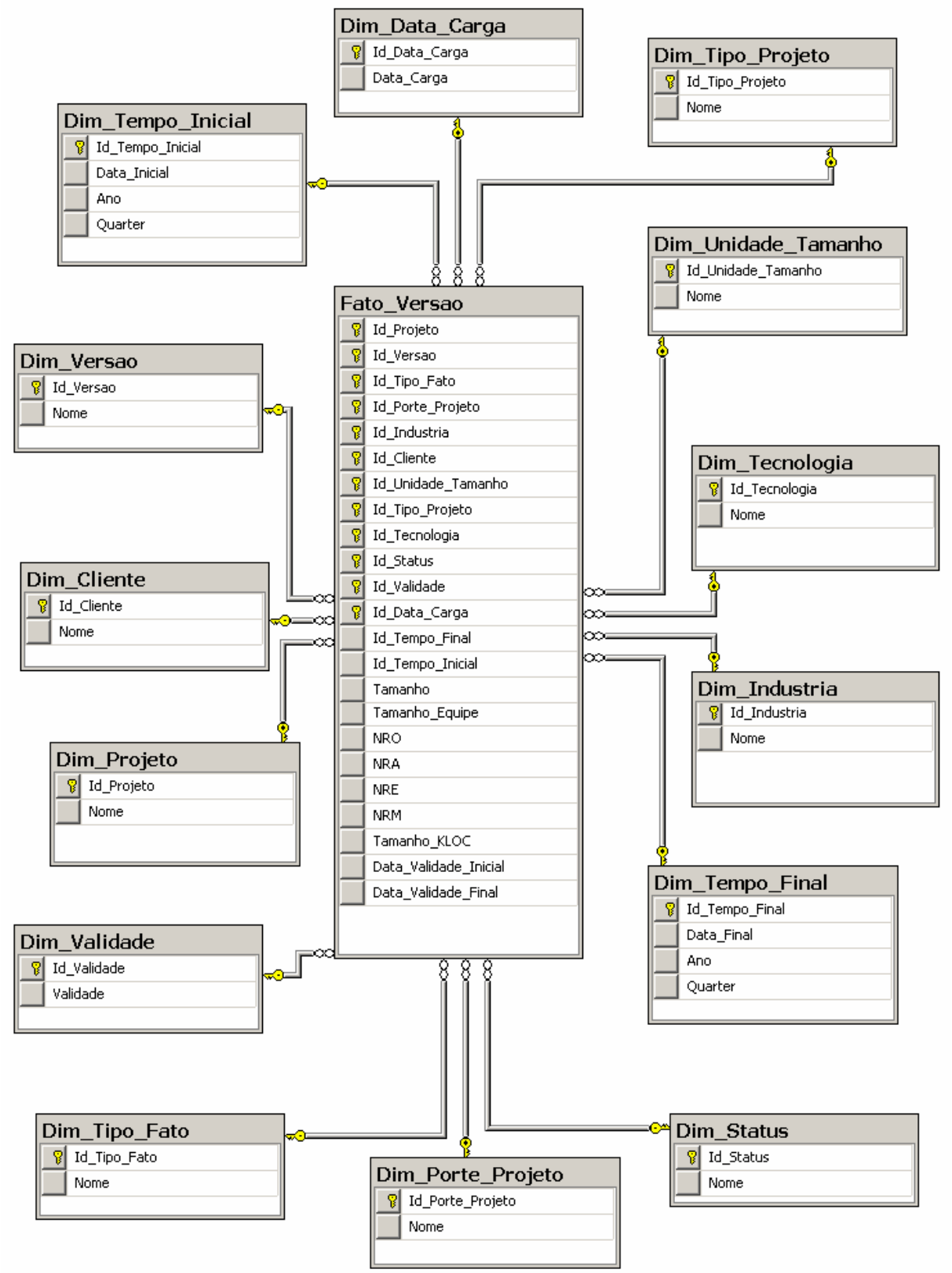
[W3C06c] W3C. **Web Service Description Language**. Disponível em: http://www.w3.org/TR/WSDL20/. Acesso em: 10 set. 2006.

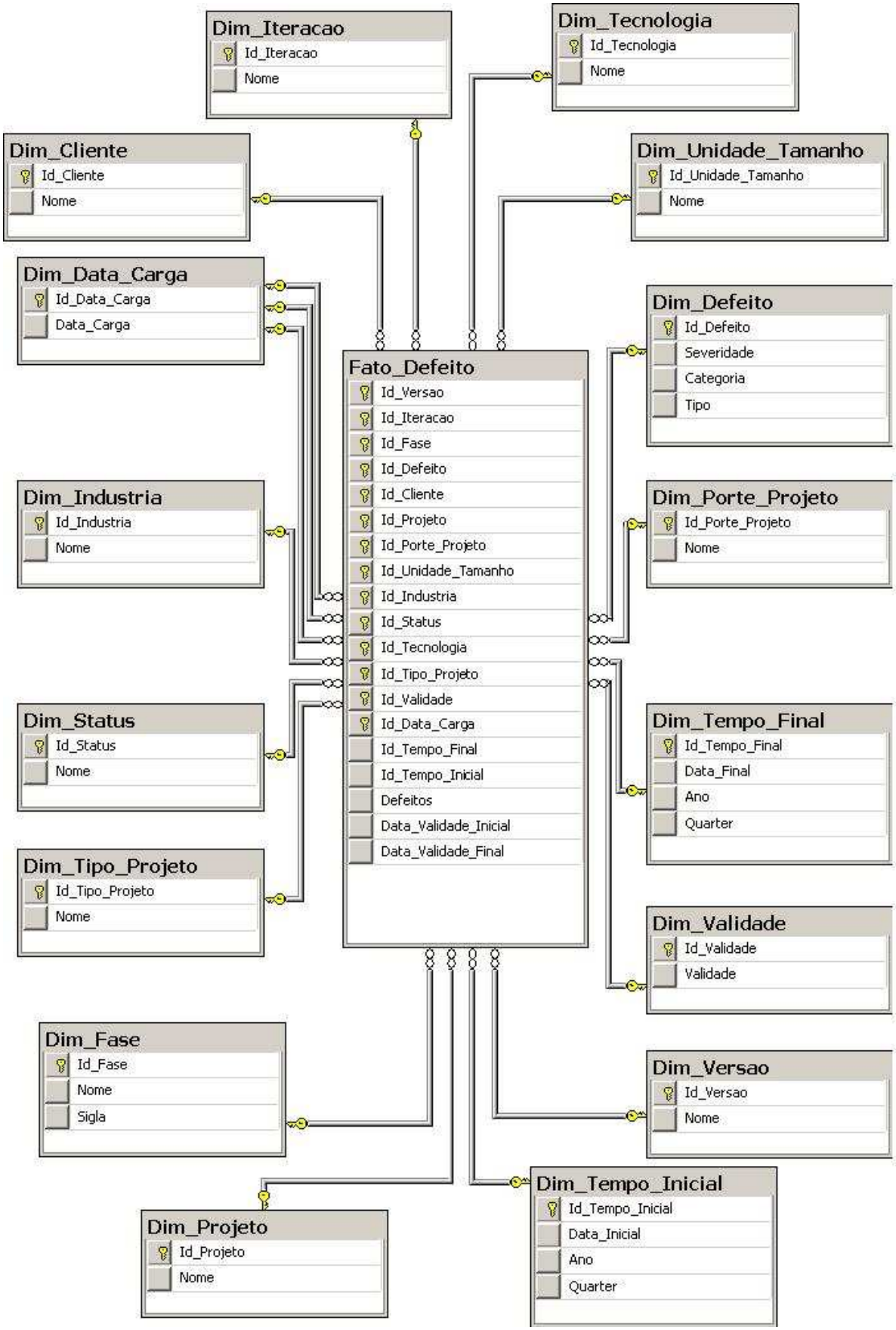
[W3C06d] W3C. **XML Schema**. Disponível em: http://www.w3.org/XML/Schema. Acesso em: 10 out. 2006.

[WON06] WONG-BUSHBYM, I.; EGAN, R.; ISAACSON, C. A Case Study in SOA and Re-Architecture at Company ABC. In: Hawaii International Conference on System Sciences, 39, 2006, Kauai, HI. **Proceedings…** Los Alamitos: IEEE Computer Society Press, 2006. p. 179b - 179b.

APÊNDICE A – Diagrama do Modelo Analítico




Dim_Data_Carga
Id_Data_Carga
Data_Carga
Dim_Tipo_Projeto
Id_Tipo_Projeto
Nome
Dim_Tempo_Inicial
Id_Tempo_Inicial
Data_Inicial
Ano
Quarter
Dim_Unidade_Tamanho
Id_Unidade_Tamanho
Nome
Fato_Versao
Id_Projeto
Id_Versao
Id_Tipo_Fato
Id_Porte_Projeto
Id_Industria
Id_Cliente
Id_Unidade_Tamanho
Id_Tipo_Projeto
Id_Tecnologia
Id_Status
Id_Validade
Id_Data_Carga
Id_Tempo_Final
Id_Tempo_Inicial
Tamanho
Tamanho_Equipe
NRO
NRA
NRE
NRM
Tamanho_KLOC
Data_Validade_Inicial
Data_Validade_Final
Dim_Versao
Id_Versao
Nome
Dim_Tecnologia
Id_Tecnologia
Nome
Dim_Cliente
Id_Cliente
Nome
Dim_Industria
Id_Industria
Nome
Dim_Projeto
Id_Projeto
Nome
Dim_Tempo_Final
Id_Tempo_Final
Data_Final
Ano
Quarter
Dim_Validade
Id_Validade
Validade
Dim_Tipo_Fato
Id_Tipo_Fato
Nome
Dim_Status
Id_Status
Nome
Dim_Porte_Projeto
Id_Porte_Projeto
Nome


Dim_Iteracao
Id_Iteracao
Nome
Dim_Tecnologia
Id_Tecnologia
Nome
Dim_Cliente
Id_Cliente
Nome
Dim_Unidade_Tamanho
Id_Unidade_Tamanho
Nome
Dim_Data_Carga
Id_Data_Carga
Data_Carga
Dim_Defeito
Id_Defeito
Severidade
Categoria
Tipo
Fato_Defeito
Id_Versao
Id_Iteracao
Id_Fase
Id_Defeito
Id_Cliente
Id_Projeto
Id_Porte_Projeto
Id_Unidade_Tamanho
Id_Industria
Id_Status
Id_Tecnologia
Id_Tipo_Projeto
Id_Validade
Id_Data_Carga
Id_Tempo_Final
Id_Tempo_Inicial
Defeitos
Data_Validade_Inicial
Data_Validade_Final
Dim_Industria
Id_Industria
Nome
Dim_Porte_Projeto
Id_Porte_Projeto
Nome
Dim_Status
Id_Status
Nome
Dim_Tempo_Final
Id_Tempo_Final
Data_Final
Ano
Quarter
Dim_Tipo_Projeto
Id_Tipo_Projeto
Nome
Dim_Validade
Id_Validade
Validade
Dim_Fase
Id_Fase
Nome
Sigla
Dim_Versao
Id_Versao
Nome
Dim_Tempo_Inicial
Id_Tempo_Inicial
Data_Inicial
Ano
Quarter
Dim_Projeto
Id_Projeto
Nome

APÊNDICE B – Metadados do Projeto1


```
<?xml version="1.0" standalone="yes" ?>
- <Projeto>
- <Ferramentas>
  <Nome>MSProject</Nome>
  <URL>jdbc:odbc:ProjectServer</URL>
  <Login>mestre</Login>
  <Senha>mestre2007</Senha>
- <Atributo>
  <Descricao>Nome_Projeto</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseText3</Campo>
  <Tipo>ntext</Tipo>
  <Tamanho>16</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Versao</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseText2</Campo>
  <Tipo>ntext</Tipo>
  <Tamanho>16</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Cliente</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseText1</Campo>
  <Tipo>ntext</Tipo>
  <Tamanho>16</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Tamanho_Time</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseNumber4</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Custo_Total</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseNumber6</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Industria</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseOutlineCode1ID</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Tecnologia</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseOutlineCode2ID</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
```

```
    </Atributo>
- <Atributo>
  <Descricao>Tipo_Projeto</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseOutlineCode3ID</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Porte_Projeto</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseOutlineCode4ID</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Fase</Descricao>
  <Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
  <Campo>TaskName</Campo>
  <Tipo>nvarchar</Tipo>
  <Tamanho>256</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Status</Descricao>
  <Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
  <Campo>TaskPercentWorkComplete</Campo>
  <Tipo>smallint</Tipo>
  <Tamanho>2</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Atividade</Descricao>
  <Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
  <Campo>TaskName</Campo>
  <Tipo>nvarchar</Tipo>
  <Tamanho>256</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Tipo_Atividade</Descricao>
  <Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
  <Campo>TaskName</Campo>
  <Tipo>nvarchar</Tipo>
  <Tamanho>256</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Custo_BO</Descricao>
  <Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
  <Campo>TaskBaselineCost</Campo>
  <Tipo>decimal</Tipo>
  <Tamanho>13</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Custo_BR</Descricao>
  <Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
  <Campo>TaskBaseline1Cost</Campo>
  <Tipo>decimal</Tipo>
  <Tamanho>13</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Custo_Real</Descricao>
```

```
<Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
<Campo>TaskCost</Campo>
<Tipo>decimal</Tipo>
<Tamanho>13</Tamanho>
</Atributo>
- <Atributo>
<Descricao>Esforco_BO</Descricao>
<Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
<Campo>TaskBaselineWork</Campo>
<Tipo>decimal</Tipo>
<Tamanho>13</Tamanho>
</Atributo>
- <Atributo>
<Descricao>Esforco_BR</Descricao>
<Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
<Campo>TaskBaseline1Work</Campo>
<Tipo>decimal</Tipo>
<Tamanho>13</Tamanho>
</Atributo>
- <Atributo>
<Descricao>Esforco_Real</Descricao>
<Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
<Campo>Esforco_Real</Campo>
<Tipo>decimal</Tipo>
<Tamanho>13</Tamanho>
</Atributo>
- <Atributo>
<Descricao>Data_Inicial_BO</Descricao>
<Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
<Campo>TaskBaselineStart</Campo>
<Tipo>datetime</Tipo>
<Tamanho>8</Tamanho>
</Atributo>
- <Atributo>
<Descricao>Data_Inicial_BR</Descricao>
<Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
<Campo>TaskBaseline1Start</Campo>
<Tipo>datetime</Tipo>
<Tamanho>8</Tamanho>
</Atributo>
- <Atributo>
<Descricao>Data_Inicial_Real</Descricao>
<Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
<Campo>TaskStart</Campo>
<Tipo>datetime</Tipo>
<Tamanho>8</Tamanho>
</Atributo>
- <Atributo>
<Descricao>Data_Final_BO</Descricao>
<Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
<Campo>TaskBaselineFinish</Campo>
<Tipo>datetime</Tipo>
<Tamanho>8</Tamanho>
</Atributo>
- <Atributo>
<Descricao>Data_Final_Real</Descricao>
<Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
<Campo>TaskFinish</Campo>
<Tipo>datetime</Tipo>
<Tamanho>8</Tamanho>
```

```
    </Atributo>
- <Atributo>
  <Descricao>Unidade_Tamanho</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseOutlineCode5ID</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Tamanho_Estimado</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseNumber1D</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Tamanho_Revisado</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseNumber3</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Tamanho_Atual</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseNumber2</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Tamanho_Atual</Descricao>
  <Tabela>MSP_VIEW_PROJ_PROJECTS_ENT</Tabela>
  <Campo>ProjectEnterpriseNumber2</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>Demanda</Descricao>
  <Tabela>MSP_VIEW_PROJ_TASKS_CF</Tabela>
  <Campo>TaskText15</Campo>
  <Tipo>ntext</Tipo>
  <Tamanho>16</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>BCWP</Descricao>
  <Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
  <Campo>TaskBCWP</Campo>
  <Tipo>decimal</Tipo>
  <Tamanho>13</Tamanho>
    </Atributo>
- <Atributo>
  <Descricao>BCWS</Descricao>
  <Tabela>MSP_VIEW_PROJ_TASKS_STD</Tabela>
  <Campo>TaskBCWS</Campo>
  <Tipo>decimal</Tipo>
  <Tamanho>13</Tamanho>
    </Atributo>
    </Ferramentas>
- <Ferramentas>
  <Nome>ClearQuest</Nome>
```

```
<URL>jdbc:odbc:ClearQuest</URL>
<Login>mestre</Login>
<Senha>mestre2007</Senha>
- <Atributo>
<Descricao>Versao</Descricao>
<Tabela>Defect</Tabela>
<Campo>versao_siapv</Campo>
<Tipo>varchar</Tipo>
<Tamanho>50</Tamanho>
    </Atributo>
- <Atributo>
<Descricao>Fase</Descricao>
<Tabela>Defect</Tabela>
<Campo>fase_origem</Campo>
<Tipo>varchar</Tipo>
<Tamanho>50</Tamanho>
    </Atributo>
- <Atributo>
<Descricao>Tipo</Descricao>
<Tabela>Defect</Tabela>
<Campo>severidade</Campo>
<Tipo>int</Tipo>
<Tamanho>4</Tamanho>
    </Atributo>
    </Ferramentas>
- <Ferramentas>
<Nome>MSExcel</Nome>
<URL>D:\Inetpub\wwwroot\MSExcel\</URL>
- <Atributo>
<Descricao>projeto</Descricao>
<Tabela>xxx</Tabela>
<Campo>xxx</Campo>
<Tipo>varchar</Tipo>
<Tamanho>50</Tamanho>
    </Atributo>
- <Atributo>
<Descricao>versao</Descricao>
<Tabela>xxx</Tabela>
<Campo>xxx</Campo>
<Tipo>varchar</Tipo>
<Tamanho>50</Tamanho>
    </Atributo>
- <Atributo>
<Descricao>Original</Descricao>
<Tabela>xxx</Tabela>
<Campo>xxx</Campo>
<Tipo>int</Tipo>
<Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
<Descricao>Adicionado</Descricao>
<Tabela>xxx</Tabela>
<Campo>xxx</Campo>
<Tipo>int</Tipo>
<Tamanho>4</Tamanho>
    </Atributo>
- <Atributo>
<Descricao>Modificado</Descricao>
<Tabela>xxx</Tabela>
<Campo>xxx</Campo>
```

```
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
     </Atributo>
- <Atributo>
  <Descricao>Eliminado</Descricao>
  <Tabela>xxx</Tabela>
  <Campo>xxx</Campo>
  <Tipo>int</Tipo>
  <Tamanho>4</Tamanho>
     </Atributo>
     </Ferramentas>
     </Projeto>
```

APÊNDICE C – DSA

```
/****************************************************************************/
/*                              D I M E N S Õ E S                           */
/****************************************************************************/
USE [DSA]
GO

/** Object:  Table [dbo].[Dim_Atividade]    Script Date: 10/04/2007
17:42:53 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Atividade](
      [Id_Atividade] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
      [Tipo] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Atividade] PRIMARY KEY CLUSTERED
(
      [Id_Atividade] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Cliente]    Script Date: 10/04/2007 17:48:37
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Cliente](
      [Id_Cliente] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Cliente] PRIMARY KEY CLUSTERED
(
      [Id_Cliente] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Data_Carga]    Script Date: 10/04/2007
17:49:35 ***/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Dim_Data_Carga](
      [Id_Data_Carga] [int] IDENTITY(1,1) NOT NULL,
      [Data_Carga] [datetime] NOT NULL,
 CONSTRAINT [PK_Dim_Data_Carga] PRIMARY KEY CLUSTERED
```

```
(
	[Id_Data_Carga] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

/** Object:  Table [dbo].[Dim_Defeito]    Script Date: 10/04/2007 17:50:02
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Defeito](
	[Id_Defeito] [int] IDENTITY(1,1) NOT NULL,
	[Severidade] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
	[Categoria] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
	[Tipo] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
 CONSTRAINT [PK_Dim_Defeito] PRIMARY KEY CLUSTERED
(
	[Id_Defeito] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Fase]    Script Date: 10/04/2007 17:50:23
*********/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Dim_Fase](
	[Id_Fase] [int] IDENTITY(1,1) NOT NULL,
	[Nome] [nchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
	[Sigla] [nchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
 CONSTRAINT [PK_Dim_Fase] PRIMARY KEY CLUSTERED
(
	[Id_Fase] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

/** Object:  Table [dbo].[Dim_Industria]    Script Date: 10/04/2007
17:50:57 ****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Industria](
	[Id_Industria] [int] IDENTITY(1,1) NOT NULL,
	[Nome] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Industria] PRIMARY KEY CLUSTERED
(
	[Id_Industria] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
```

```
GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Iteracao]    Script Date: 10/04/2007 17:51:23
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Iteracao](
      [Id_Iteracao] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Iteracao] PRIMARY KEY CLUSTERED
(
      [Id_Iteracao] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Porte_Projeto]    Script Date: 10/04/2007
17:51:42*/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Porte_Projeto](
      [Id_Porte_Projeto] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Porte_Projeto] PRIMARY KEY CLUSTERED
(
      [Id_Porte_Projeto] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Projeto]    Script Date: 10/04/2007 17:52:15
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Projeto](
      [Id_Projeto] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Projeto] PRIMARY KEY CLUSTERED
(
      [Id_Projeto] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
```

```
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Status]    Script Date: 10/04/2007 17:52:41
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Status](
      [Id_Status] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Status] PRIMARY KEY CLUSTERED
(
      [Id_Status] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Tecnologia]    Script Date: 10/04/2007
17:53:06 ***/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Tecnologia](
      [Id_Tecnologia] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Tecnologia] PRIMARY KEY CLUSTERED
(
      [Id_Tecnologia] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Tempo_Final]    Script Date: 10/04/2007
17:53:19 **/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Dim_Tempo_Final](
      [Id_Tempo_Final] [int] IDENTITY(1,1) NOT NULL,
      [Data_Final] [datetime] NULL,
      [Ano] [int] NULL,
      [Quarter] [int] NULL,
 CONSTRAINT [PK_Dim_Tempo_Final] PRIMARY KEY CLUSTERED
(
      [Id_Tempo_Final] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

/** Object:  Table [dbo].[Dim_Tempo_Inicial]    Script Date: 10/04/2007
17:53:31*/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Dim_Tempo_Inicial](
      [Id_Tempo_Inicial] [int] IDENTITY(1,1) NOT NULL,
      [Data_Inicial] [datetime] NULL,
      [Ano] [int] NULL,
      [Quarter] [int] NULL,
 CONSTRAINT [PK_Dim_Tempo_Inicial] PRIMARY KEY CLUSTERED
(
      [Id_Tempo_Inicial] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

/** Object:  Table [dbo].[Dim_Tipo_Fato]    Script Date: 10/04/2007
17:53:42 ****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Tipo_Fato](
      [Id_Tipo_Fato] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Tipo_Fato] PRIMARY KEY CLUSTERED
(
      [Id_Tipo_Fato] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Tipo_Projeto]    Script Date: 10/04/2007
17:53:55 */
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Tipo_Projeto](
      [Id_Tipo_Projeto] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Tipo_Projeto] PRIMARY KEY CLUSTERED
(
      [Id_Tipo_Projeto] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Unidade_Tamanho] Script Date: 10/04/2007
17:54:07 */
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
```

```
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Unidade_Tamanho](
      [Id_Unidade_Tamanho] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](20) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Unidade_Tamanho] PRIMARY KEY CLUSTERED
(
      [Id_Unidade_Tamanho] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[Dim_Versao]    Script Date: 10/04/2007 17:54:25
*******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Dim_Versao](
      [Id_Versao] [int] IDENTITY(1,1) NOT NULL,
      [Nome] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
 CONSTRAINT [PK_Dim_Versao] PRIMARY KEY CLUSTERED
(
      [Id_Versao] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/*************************************************************************/
/*         T A B E L A S      A U X I L I A R E S     D A    D S A       */
/*************************************************************************
/** Object:  Table [dbo].[DSA_Atividade]    Script Date: 10/04/2007
17:59:20 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[DSA_Atividade](
      [Id_Projeto] [int] NULL,
      [Id_Atividade] [int] NULL,
      [Id_Versao] [int] NULL,
      [Id_Iteracao] [int] NULL,
      [Id_Fase] [nchar](10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
      [Nome_Projeto] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT
NULL,
      [Nome_Atividade] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS
NULL,
      [Versao] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
      [Iteracao] [int] NULL,
      [Fase] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
      [Tipo] [varchar](30) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
      [Data_Inicial_Real] [datetime] NULL,
      [Data_Final_Real] [datetime] NULL,
```

```
	[Data_Inicial_BO] [datetime] NULL,
	[Data_Final_BO] [datetime] NULL,
	[Data_Inicial_BR] [datetime] NULL,
	[Data_Final_BR] [datetime] NULL,
	[Esforco_Real] [real] NULL,
	[Esforco_Original] [real] NULL,
	[Esforco_Revisado] [real] NULL,
	[Custo_Real] [real] NULL,
	[Custo_Original] [real] NULL,
	[Custo_Revisado] [real] NULL,
	[Work_Completed] [real] NULL,
	[Data_Carga] [datetime] NOT NULL
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[DSA_Defeito]    Script Date: 10/04/2007 18:01:08
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[DSA_Defeito](
	[Id_Projeto] [int] NULL,
	[Id_Versao] [nchar](10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
	[Id_Iteracao] [nchar](10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
	[Id_Fase] [nchar](10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
	[Id_Defeito] [nchar](10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
	[Versao] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
	[Nome_Projeto] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT
NULL,
	[Total] [int] NOT NULL,
	[Iteracao] [int] NULL,
	[Fase] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
	[Tipo] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
	[Categoria] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT
NULL,
	[Severidade] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
	[Data_Carga] [datetime] NOT NULL,
	[Data_Abertura] [datetime] NULL,
	[Data_Fechamento] [datetime] NULL
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[DSA_Produto]    Script Date: 10/04/2007 18:01:25
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[DSA_Produto](
	[Id_Versao] [int] NULL,
	[Versao] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
```

```
	[Nome_Projeto] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT
NULL,
	[Cliente] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
	[Industria] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
	[Tecnologia] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS
NULL,
	[Tipo_Projeto] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS
NULL,
	[Porte_Projeto] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS
NULL,
	[Satisfacao_Cliente] [real] NULL,
	[Unidade_Tamanho] [varchar](20) COLLATE SQL_Latin1_General_CP1_CI_AS
NULL,
	[Status] [varchar](100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,
	[Tamanho_Equipe] [int] NULL,
	[Tamanho_Real] [real] NULL,
	[Tamanho_Original] [real] NULL,
	[Tamanho_Revisado] [real] NULL,
	[Tamanho_Total_KLOC] [real] NULL,
	[Fator_Conversao] [real] NULL,
	[Data_Carga] [datetime] NOT NULL
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/** Object:  Table [dbo].[DSA_Requisito]    Script Date: 10/04/2007
18:01:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[DSA_Requisito](
	[Id_Projeto] [int] NULL,
	[Id_Versao] [int] NULL,
	[Versao] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL,
	[Nome_Projeto] [varchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT
NULL,
	[Requisitos_Originais] [int] NULL,
	[Requisitos_Adicionados] [int] NULL,
	[Requisitos_Eliminados] [int] NULL,
	[Requisitos_Modificados] [int] NULL,
	[Data_Carga] [datetime] NULL
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF

/*****************************************************************/
/*                              F A T O S                        */
/*****************************************************************
/** Object:  Table [dbo].[DSA_Fato_Atividade]    Script Date: 10/05/2007
16:45:50 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[DSA_Fato_Atividade](
	[Id_Projeto] [int] NOT NULL,
```

```
	[Id_Versao] [int] NOT NULL,
	[Id_Iteracao] [int] NOT NULL,
	[Id_Fase] [int] NOT NULL,
	[Id_Atividade] [int] NOT NULL,
	[Id_Cliente] [int] NOT NULL,
	[Id_Porte_Projeto] [int] NOT NULL,
	[Id_Industria] [int] NOT NULL,
	[Id_Unidade_Tamanho] [int] NOT NULL,
	[Id_Tipo_Projeto] [int] NOT NULL,
	[Id_Tecnologia] [int] NOT NULL,
	[Id_Status] [int] NOT NULL,
	[Id_Data_Carga] [int] NOT NULL,
	[Id_Tipo_Fato] [int] NOT NULL,
	[Id_Tempo_Inicial] [int] NULL,
	[Id_Tempo_Final] [int] NULL,
	[Data_Inicial] [datetime] NULL,
	[Data_Final] [datetime] NULL,
	[Work_Completed] [real] NULL,
	[Esforco] [real] NULL,
	[DS] [datetime] NULL,
	[Custo] [real] NULL,
	[Data_Validade_Inicial] [datetime] NOT NULL,
	[Data_Validade_Final] [datetime] NULL,
 CONSTRAINT [PK_DSA_Fato_Atividade_1] PRIMARY KEY CLUSTERED
(
	[Id_Projeto] ASC,
	[Id_Versao] ASC,
	[Id_Iteracao] ASC,
	[Id_Fase] ASC,
	[Id_Atividade] ASC,
	[Id_Cliente] ASC,
	[Id_Porte_Projeto] ASC,
	[Id_Industria] ASC,
	[Id_Unidade_Tamanho] ASC,
	[Id_Tipo_Projeto] ASC,
	[Id_Tecnologia] ASC,
	[Id_Status] ASC,
	[Id_Data_Carga] ASC,
	[Id_Tipo_Fato] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
USE [DSA]
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Atividade]
FOREIGN KEY([Id_Atividade])
REFERENCES [dbo].[Dim_Atividade] ([Id_Atividade])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Cliente]
FOREIGN KEY([Id_Cliente])
REFERENCES [dbo].[Dim_Cliente] ([Id_Cliente])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Data_Carga]
FOREIGN KEY([Id_Data_Carga])
REFERENCES [dbo].[Dim_Data_Carga] ([Id_Data_Carga])
GO
```

```
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Fase]
FOREIGN KEY([Id_Fase])
REFERENCES [dbo].[Dim_Fase] ([Id_Fase])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Industria]
FOREIGN KEY([Id_Industria])
REFERENCES [dbo].[Dim_Industria] ([Id_Industria])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Iteracao]
FOREIGN KEY([Id_Iteracao])
REFERENCES [dbo].[Dim_Iteracao] ([Id_Iteracao])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Porte_Projeto]
FOREIGN KEY([Id_Porte_Projeto])
REFERENCES [dbo].[Dim_Porte_Projeto] ([Id_Porte_Projeto])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Projeto]
FOREIGN KEY([Id_Projeto])
REFERENCES [dbo].[Dim_Projeto] ([Id_Projeto])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Status]
FOREIGN KEY([Id_Status])
REFERENCES [dbo].[Dim_Status] ([Id_Status])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Tecnologia]
FOREIGN KEY([Id_Tecnologia])
REFERENCES [dbo].[Dim_Tecnologia] ([Id_Tecnologia])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Tempo_Final1]
FOREIGN KEY([Id_Tempo_Inicial])
REFERENCES [dbo].[Dim_Tempo_Final] ([Id_Tempo_Final])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Tempo_Inicial]
FOREIGN KEY([Id_Tempo_Inicial])
REFERENCES [dbo].[Dim_Tempo_Inicial] ([Id_Tempo_Inicial])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Tipo_Fato]
FOREIGN KEY([Id_Tipo_Fato])
REFERENCES [dbo].[Dim_Tipo_Fato] ([Id_Tipo_Fato])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Tipo_Projeto]
FOREIGN KEY([Id_Tipo_Projeto])
REFERENCES [dbo].[Dim_Tipo_Projeto] ([Id_Tipo_Projeto])
GO
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Unidade_Tamanho]
FOREIGN KEY([Id_Unidade_Tamanho])
REFERENCES [dbo].[Dim_Unidade_Tamanho] ([Id_Unidade_Tamanho])
GO
```

```
ALTER TABLE [dbo].[DSA_Fato_Atividade]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Atividade_Dim_Versao]
FOREIGN KEY([Id_Versao])
REFERENCES [dbo].[Dim_Versao] ([Id_Versao])


/** Object:  Table [dbo].[DSA_Fato_Defeito]    Script Date: 10/05/2007
16:46:37 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[DSA_Fato_Defeito](
      [Id_Versao] [int] NOT NULL,
      [Id_Iteracao] [int] NOT NULL,
      [Id_Fase] [int] NOT NULL,
      [Id_Defeito] [int] NOT NULL,
      [Id_Cliente] [int] NOT NULL,
      [Id_Projeto] [int] NOT NULL,
      [Id_Porte_Projeto] [int] NOT NULL,
      [Id_Unidade_Tamanho] [int] NOT NULL,
      [Id_Industria] [int] NOT NULL,
      [Id_Status] [int] NOT NULL,
      [Id_Tecnologia] [int] NOT NULL,
      [Id_Tipo_Projeto] [int] NOT NULL,
      [Id_Data_Carga] [int] NOT NULL,
      [Id_Tempo_Inicial] [int] NULL,
      [Id_Tempo_Final] [int] NULL,
      [Defeitos] [int] NULL,
      [Data_Validade_Inicial] [datetime] NOT NULL,
      [Data_Validade_Final] [datetime] NULL,
 CONSTRAINT [PK_DSA_Fato_Defeito_1] PRIMARY KEY CLUSTERED
(
      [Id_Versao] ASC,
      [Id_Iteracao] ASC,
      [Id_Fase] ASC,
      [Id_Defeito] ASC,
      [Id_Cliente] ASC,
      [Id_Projeto] ASC,
      [Id_Porte_Projeto] ASC,
      [Id_Unidade_Tamanho] ASC,
      [Id_Industria] ASC,
      [Id_Status] ASC,
      [Id_Tecnologia] ASC,
      [Id_Tipo_Projeto] ASC,
      [Id_Data_Carga] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
USE [DSA]
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Cliente]
FOREIGN KEY([Id_Cliente])
REFERENCES [dbo].[Dim_Cliente] ([Id_Cliente])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Data_Carga]
FOREIGN KEY([Id_Data_Carga])
REFERENCES [dbo].[Dim_Data_Carga] ([Id_Data_Carga])
```

```
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Defeito]
FOREIGN KEY([Id_Defeito])
REFERENCES [dbo].[Dim_Defeito] ([Id_Defeito])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Fase]
FOREIGN KEY([Id_Fase])
REFERENCES [dbo].[Dim_Fase] ([Id_Fase])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Industria]
FOREIGN KEY([Id_Industria])
REFERENCES [dbo].[Dim_Industria] ([Id_Industria])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Iteracao]
FOREIGN KEY([Id_Iteracao])
REFERENCES [dbo].[Dim_Iteracao] ([Id_Iteracao])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Porte_Projeto]
FOREIGN KEY([Id_Porte_Projeto])
REFERENCES [dbo].[Dim_Porte_Projeto] ([Id_Porte_Projeto])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Projeto]
FOREIGN KEY([Id_Projeto])
REFERENCES [dbo].[Dim_Projeto] ([Id_Projeto])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Status]
FOREIGN KEY([Id_Status])
REFERENCES [dbo].[Dim_Status] ([Id_Status])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Tecnologia]
FOREIGN KEY([Id_Tecnologia])
REFERENCES [dbo].[Dim_Tecnologia] ([Id_Tecnologia])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Tempo_Final]
FOREIGN KEY([Id_Tempo_Final])
REFERENCES [dbo].[Dim_Tempo_Final] ([Id_Tempo_Final])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Tempo_Inicial]
FOREIGN KEY([Id_Tempo_Inicial])
REFERENCES [dbo].[Dim_Tempo_Inicial] ([Id_Tempo_Inicial])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Tipo_Projeto]
FOREIGN KEY([Id_Tipo_Projeto])
REFERENCES [dbo].[Dim_Tipo_Projeto] ([Id_Tipo_Projeto])
GO
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Unidade_Tamanho]
FOREIGN KEY([Id_Unidade_Tamanho])
REFERENCES [dbo].[Dim_Unidade_Tamanho] ([Id_Unidade_Tamanho])
GO
```

```
ALTER TABLE [dbo].[DSA_Fato_Defeito]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Defeito_Dim_Versao]
FOREIGN KEY([Id_Versao])
REFERENCES [dbo].[Dim_Versao] ([Id_Versao])


/** Object:  Table [dbo].[DSA_Fato_Versao]    Script Date: 10/05/2007
16:47:19 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[DSA_Fato_Versao](
      [Id_Projeto] [int] NOT NULL,
      [Id_Versao] [int] NOT NULL,
      [Id_Tipo_Fato] [int] NOT NULL,
      [Id_Porte_Projeto] [int] NOT NULL,
      [Id_Industria] [int] NOT NULL,
      [Id_Cliente] [int] NOT NULL,
      [Id_Unidade_Tamanho] [int] NOT NULL,
      [Id_Tipo_Projeto] [int] NOT NULL,
      [Id_Tecnologia] [int] NOT NULL,
      [Id_Status] [int] NOT NULL,
      [Id_Data_Carga] [int] NOT NULL,
      [Id_Tempo_Inicial] [int] NULL,
      [Id_Tempo_Final] [int] NULL,
      [Tamanho] [real] NULL,
      [Tamanho_Equipe] [int] NULL,
      [NRO] [int] NULL,
      [NRA] [int] NULL,
      [NRE] [int] NULL,
      [NRM] [int] NULL,
      [Tamanho_KLOC] [real] NULL,
      [Data_Validade_Inicial] [datetime] NOT NULL,
      [Data_Validade_Final] [datetime] NULL,
 CONSTRAINT [PK_DSA_Fato_Versao_1] PRIMARY KEY CLUSTERED
(
      [Id_Projeto] ASC,
      [Id_Versao] ASC,
      [Id_Tipo_Fato] ASC,
      [Id_Porte_Projeto] ASC,
      [Id_Industria] ASC,
      [Id_Cliente] ASC,
      [Id_Unidade_Tamanho] ASC,
      [Id_Tipo_Projeto] ASC,
      [Id_Tecnologia] ASC,
      [Id_Status] ASC,
      [Id_Data_Carga] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
USE [DSA]
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Cliente]
FOREIGN KEY([Id_Cliente])
REFERENCES [dbo].[Dim_Cliente] ([Id_Cliente])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Data_Carga]
```

```
FOREIGN KEY([Id_Data_Carga])
REFERENCES [dbo].[Dim_Data_Carga] ([Id_Data_Carga])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Industria]
FOREIGN KEY([Id_Industria])
REFERENCES [dbo].[Dim_Industria] ([Id_Industria])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Porte_Projeto]
FOREIGN KEY([Id_Porte_Projeto])
REFERENCES [dbo].[Dim_Porte_Projeto] ([Id_Porte_Projeto])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Projeto]
FOREIGN KEY([Id_Projeto])
REFERENCES [dbo].[Dim_Projeto] ([Id_Projeto])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Status]
FOREIGN KEY([Id_Status])
REFERENCES [dbo].[Dim_Status] ([Id_Status])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Tecnologia]
FOREIGN KEY([Id_Tecnologia])
REFERENCES [dbo].[Dim_Tecnologia] ([Id_Tecnologia])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Tempo_Final]
FOREIGN KEY([Id_Tempo_Final])
REFERENCES [dbo].[Dim_Tempo_Final] ([Id_Tempo_Final])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Tempo_Inicial]
FOREIGN KEY([Id_Tempo_Inicial])
REFERENCES [dbo].[Dim_Tempo_Inicial] ([Id_Tempo_Inicial])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Tipo_Fato]
FOREIGN KEY([Id_Tipo_Fato])
REFERENCES [dbo].[Dim_Tipo_Fato] ([Id_Tipo_Fato])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Tipo_Projeto]
FOREIGN KEY([Id_Tipo_Projeto])
REFERENCES [dbo].[Dim_Tipo_Projeto] ([Id_Tipo_Projeto])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Unidade_Tamanho]
FOREIGN KEY([Id_Unidade_Tamanho])
REFERENCES [dbo].[Dim_Unidade_Tamanho] ([Id_Unidade_Tamanho])
GO
ALTER TABLE [dbo].[DSA_Fato_Versao]  WITH CHECK ADD  CONSTRAINT
[FK_DSA_Fato_Versao_Dim_Versao]
FOREIGN KEY([Id_Versao])
REFERENCES [dbo].[Dim_Versao] ([Id_Versao])


/************************************************************************/
```

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo