# Experiência em Processo de Teste Iterativo e Automatizado para Data Warehouse

**Luana M. de A. Lobão[1], Eliane F. Collins[1,2], , Ronaldo N. Martins[1], Leandro A. O. Leão[1], Vicente Ferreira de Lucena Jr.[2]**

[1]Instituto Nokia de Tecnologia (INdT)
Caixa Postal 7200 – 69048-660 – Manaus – AM – Brasil
[2]Universidade Federal do Amazonas (UFAM)
Caixa Postal 1200 – 69065-020 – Manaus – AM – Brasil

{eliane.collins, leandro.leao, luana.lobao, ronaldo.martins}@indt.org.br, vicente@ufam.edu.br

***Abstract.*** *This paper describes the experience of developing and implementing a testing process for Data Warehouse in order to quality control of important information in databases used as analytical support for decisions. During the development of a Data Warehouse, the data is manipulated in other data sources and possibly rescaled, if an error occurs during this process the resulting information will be compromised. In order to reduce this risk, a test process was proposed, implemented and executed, the results obtained showed the benefits of using testing techniques to prevent defects, reducing costs and ensuring the correct information integrity.*

***Resumo.*** *Este artigo descreve a experiência de desenvolver e aplicar um processo de teste para Data Warehouse visando o controle da qualidade de informações importantes em bancos de dados analíticos que servem de apoio às decisões. Durante o desenvolvimento de um Data Warehouse, os dados são manipulados de outras fontes de dados e possivelmente renormalizados, se ocorrer um erro durante esse processo as informações resultantes estarão comprometidas. Para reduzir esse risco, um processo de teste foi proposto, implementado e executado, os resultados obtidos mostraram os benefícios de utilizar técnicas de teste para prevenção de defeitos, reduzindo custos com correção e garantindo a integridade da informação.*

## 1. Introdução

Um dos ativos mais importantes em uma organização é a sua informação. A informação é adquirida usualmente através de sistemas de registro e armazenamento de dados e para ser disponibilizada de maneira segura e eficiente têm-se feito uso de *Data Warehouses* (DW). Isso se deve principalmente à necessidade de tornar a informação acessível, consistente, flexível e confiável o suficiente para apoiar decisões [Kimball 2002].

O processo de desenvolvimento de um DW envolve a manipulação de várias bases de dados de sistemas legados (*Data Sources*), esses dados são tratados em procedimentos chamados *extract-transformation-load (ETL)* que transformam, combinam, duplicam e arquivam os dados para serem usados posteriormente. A modelagem dos dados é chamada de Modelo Dimensional. Este modelo contém os relacionamentos das entidades em pacotes formatados para serem bem compreendidos.

Os principais componentes do Modelo Dimensional são as Tabelas de Fato, que correspondem a tabelas primárias. Estas contêm as métricas do negócio e são caracterizadas por cada Dimensão do modelo. Um elemento interessante de um DW é o *Data Mart*, que corresponde ao subconjunto categorizado de dados referentes a um assunto específico (por exemplo vendas e estoque) e suportam intensa pesquisa. Contudo, para possibilitar a análise das dimensões desses dados utilizam-se ferramentas OLAP (*Online Analytical Processing*) que são usadas por gestores de organizações como apoio a decisões [Kimball 2002].

Como pode ser observada, a complexidade que envolve a criação e manipulação de dados de um DW requer cuidado em todos os níveis do seu processo. Cometer um erro em um procedimento ou na modelagem pode custar muito caro para a organização que depende desses dados. Pode-se concluir que no processo de desenvolvimento de um DW a execução de testes deve ter uma devida importância. Com essa perspectiva, neste artigo é descrito a aplicação de uma proposta de processo de teste que visa atender à necessidade de garantir qualidade da informação obtida de um DW. Na seção 2 há informações de como era feito o desenvolvimento do DW na organização e os problemas identificados. Na seção 3, o projeto piloto utilizado para a experiência e na seção 4, o processo de teste proposto. Na seção 5 os resultados obtidos com a experiência são comentados e por fim na seção 6, as lições aprendidas e os projetos futuros para a evolução do processo de teste.

## 2. Desenvolvimento de DW na Organização

O Instituto Nokia de Tecnologia (INdT) é uma instituição independente e sem fins lucrativos. É comprometida com a realização de pesquisa e desenvolvimento de soluções tecnológicas através do desenvolvimento de aplicações, novas tecnologias e conceitos. As principais áreas do INdT são Software Livre e Interfaces de Usuário, Tecnologias de Produto e Manufatura, Experiências em Serviços e Tecnologias de Rede. A área em que a experiência foi desenvolvida foi Tecnologias de Produto e Manufatura [Collins e Lobão 2010].

O desenvolvimento de DW no Instituto era feito de acordo com o fluxo ilustrado na Figura 1. Para a construção de um *Data Mart* era utilizada a ferramenta IBM Cognos® [IBM Cognos 2010], que é uma plataforma profissional de desenvolvimento de sistemas de *Business Intelligence (BI)*. A modelagem Dimensional usada nos projetos para a construção desta base analítica é a proposta por Kimball [Kimball 2002]. Após a construção da base analítica era feito todo o processo de ETL. Neste processo os dados eram trabalhados e migrados da base legada para uma base intermediária (chamada de *stage*) e desta base intermediária para o *Data Mart*.

Toda a verificação e validação de dados era feita apenas na etapa final do desenvolvimento, ou seja, na fase de relatórios. Nesta fase toda a migração de dados já foi completada sem ser validada. Com isso a identificação de problemas era tardia e demandava um esforço muito grande para checar que partes do sistema estavam gerando aqueles erros. Isso causava um esforço muito grande para a conclusão do projeto, gerando uma enorme quantidade de retrabalho para a equipe. O processo de teste que será mostrado neste artigo foi proposto para reduzir os problemas identificados nas experiências anteriores pelos desenvolvedores na construção de DW.

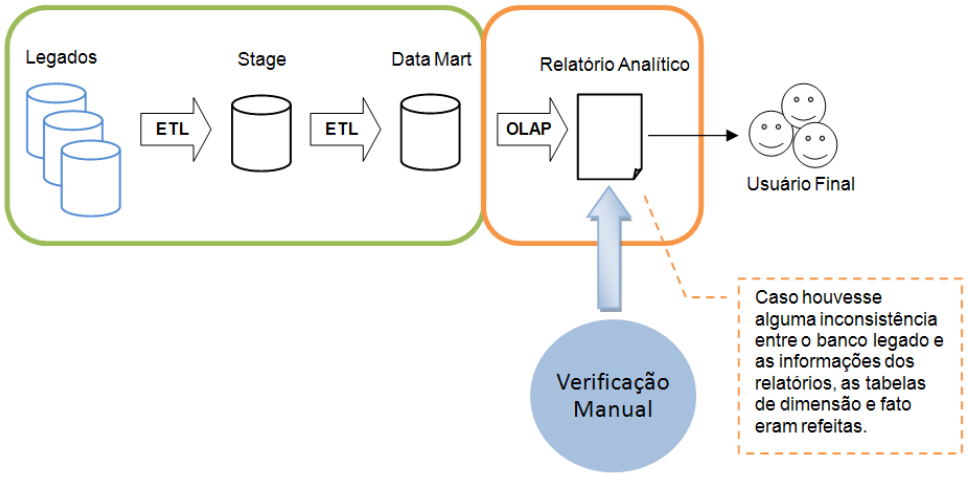



**Figura 1 – Execução de Testes Manuais**

## 3. Projeto Piloto

O Projeto Piloto que serviu para a aplicação do processo de testes tinha como principal objetivo reunir e organizar dados sobre o ciclo de vida de equipamentos de uma empresa de eletroeletrônicos. Após esta organização, estes dados eram transformados em informações estratégicas relacionadas à manutenção destes equipamentos. Portanto, havia um sistema que era responsável apenas por guardar e manter os dados dos equipamentos em uma base. Esta base foi chamada de "legado". Com isso foi construído um *data mart* que continha os dados do legado reorganizados de maneira a facilitar a geração de relatórios estratégicos. Para sua construção a equipe continuou usando a ferramenta IBM Cognos® [IBM Cognos 2010].

O Projeto Piloto utilizou *Scrum* como metodologia de desenvolvimento [SCHWABER 2004]. Murphy descreve o *Scrum* como um processo de gestão de software e desenvolvimento de produtos que utiliza iterações e práticas incrementais, para produzir produtos que agregam valor ao negócio, podendo ser também aplicada a projetos de outras naturezas, bem como à gestão de programas [MURPHY 2004]. No desenvolvimento de software na organização, a equipe de teste faz parte do time *Scrum*, portanto, participa ativamente de todas as cerimônias do framework.

A Figura 2 apresenta o ciclo de vida de projetos sob a metodologia *Scrum*, em que o objetivo, escopo de projetos e requisitos são definidos como estórias do cliente. Estas estórias são levantadas com uma forte participação do cliente e registradas em um documento chamado "*Product Backlog*". Este armazena as funcionalidades requeridas pelo usuário. O *Backlog* de Produto é alterado no decorrer do projeto de acordo com a necessidade, sendo usado como entrada básica para o planejamento da execução de iterações de desenvolvimento [Leão e Quaglia, 2010].


**Figura 2 – Ciclo de Vida no *SCRUM* (apud Leão e Quaglia, 2010)**

## 4. Processo de Teste Proposto para DW

O processo de teste elaborado para validar as etapas de desenvolvimento de um DW é ilustrado na Figura 3. Este processo foi desenhado com base nas fases de desenvolvimento do modelo físico do DW. Entenda por fases de desenvolvimento, a construção de Tabelas de Dimensão e Fato e o processo ETL que gira em torno do processo de migração de dados. Em um DW, a integridade dos dados entre as bases (legado, *stage* e *Data Mart*) precisa ser garantida.

De acordo com Kimball e Ross (2002), um dos principais objetivos de um DW é apresentar uma informação concisa baseada em dados íntegros e confiáveis. Portanto a cada desenvolvimento de *scripts* SQL com propósito de ETL entre as bases, o processo de teste era executado para garantir essa integridade de dados entre bases diferentes. Também se fez uso de área de *stage* em que são feitas réplicas exatas das tabelas de dados dos sistemas legados da empresa de forma temporária para futuro processamento no ETL. O objetivo era não sobrecarregar as bases de dados legadas com operações de ordenação, agregações e junções e não afetar seus sistemas críticos.

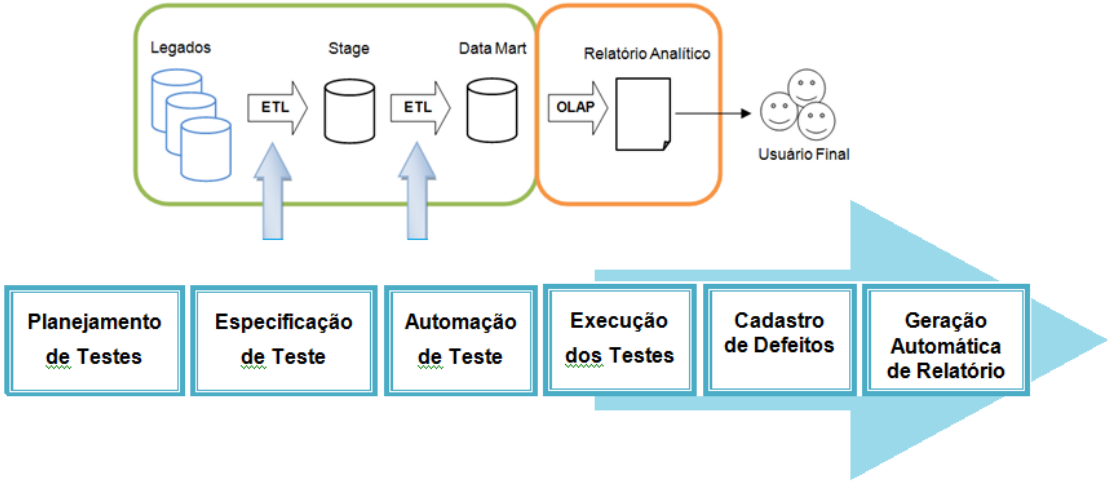



**Figura 3 – Processo de Teste Iterativo**

Na fase de Planejamento de Testes, a equipe de teste se reuniu para entender o processo de desenvolvimento do DW. O planejamento dos testes foi feito levando em

consideração a experiência anterior dos desenvolvedores em construção de um DW. Este contato com os desenvolvedores foi muito importante, pois assim a equipe de qualidade conseguiu entender e identificar aonde os principais esforços de teste deveriam se concentrar. Com isso foram definidos os seguintes pontos fortes a serem testados e verificados:

a) Verificar o mapeamento dos campos presentes entre as áreas de legado/*stage*;

b) Verificar se houve valores duplicados no processo de ETL de carga de dados entre as bases (do Legado para *stage* e do *stage* para o *Data Mart*);

c) Verificar os tipos de dados dos campos correspondentes nos diferentes níveis (Legado, *stage* e *Data mart*);

d) Verificar a quantidade de dados no processo de ETL de carga de dados entre as bases (do Legado para *stage* e do *stage* para o *Data mart*);

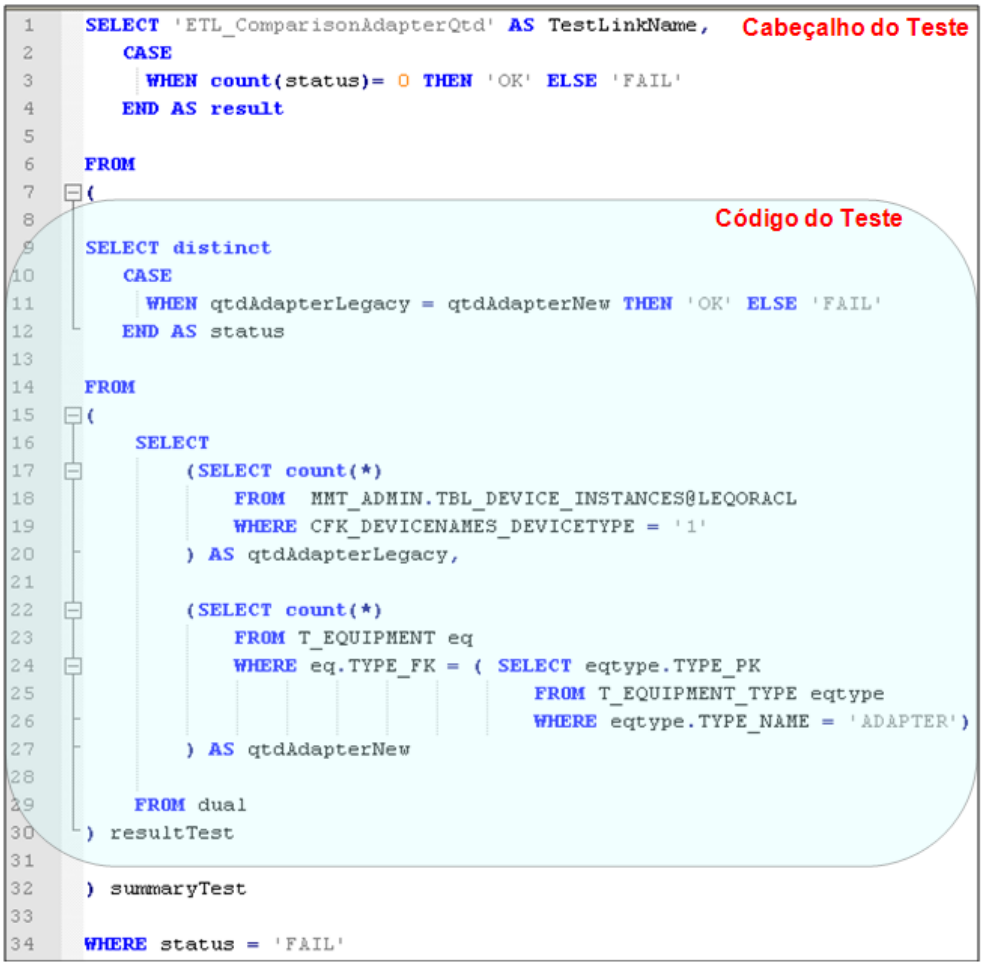
e) Verificar se os *Jobs* (Scripts que controlam a execução da carga, tanto na stage quanto no *Data mart*) estavam funcionando de acordo com a especificação, ou seja, se iniciavam no tempo pré-configurado.

Entendendo o que seria testado e como, a equipe de teste criou a Especificação de Teste. Este documento basicamente descrevia todos os possíveis casos de teste levantados com base nos pontos fortes a serem testados descritos acima e no processo iterativo de desenvolvimento. Estes casos de teste eram constantemente atualizados de acordo com a iteração de desenvolvimento corrente, ou seja, a cada Tabela de Fato e Dimensão criada e posteriormente com a criação de *Jobs*. Todo o processo de desenvolvimento e teste foi executado seguindo as iterações e cerimônias do processo *Scrum*. Isso facilitou o trabalho e ajudou bastante a equipe de qualidade, pois o processo de teste executado foi criado a partir de elementos da metodologia ágil *Scrum* [Collins e Lobão 2010]. Portanto, a fase de planejamento, especificação e execução de testes encaixou perfeitamente com a natureza iterativa do processo de desenvolvimento do DW. O software utilizado para o gerenciamento de casos de testes foi o TestLink [TestLink 2009], que é uma ferramenta *Open Source* bastante utilizada e difundida entre profissionais de teste de software.

Por se tratar de um sistema que está sempre lidando com uma enorme quantidade de dados, desde o início a automação dos casos de teste era prevista. Além de ajudar a definir os cenários de teste, a conversa com a equipe de desenvolvimento, ajudou a diagnosticar os casos que poderiam ser automatizados. A ferramenta utilizada para a automação foi o *Data Manager* [IBM Cognos 2010], mesma ferramenta utilizada pelos desenvolvedores para construir o DW. Dos pontos fortes levantados, apenas três (letras “b”, “c” e “d”) foram efetivamente automatizados inteiramente pela ferramenta. Isso quer dizer que até a geração do resultado da execução foi automatizada. O passo “a” foi verificado manualmente através de um recurso visual do *Data Manager* na construção do ETL. O passo “e” foi executado e verificado manualmente com a ajuda do PL/SQL [Oracle 2010], que é um Sistema Gerenciador de Banco de Dados.

Para a automação dos passos “b”, “c” e “d” foram basicamente feitos *scripts* SQL, que checavam valores duplicados, tipos de campos e quantidade de registros. A cada objeto de negócio feito no *Data Manager* para construir Tabelas de Fato, de Dimensão e fazer carga e migração de dados, era feito um objeto de teste

correspondente. Este objeto de teste continha instruções SQL que manipulavam as mesmas bases de dados que os objetos de negócio. Estes scripts SQL continham instruções de Select e Insert na maioria das vezes. Na Figura 4 pode ser visto um exemplo de script que comparava a quantidade de dados de uma base de negócio (*MMT_ADMIN.TBL_DEVICE_INSTANCES@LEQORACL*) com uma base de *stage* (*Data mart - T_EQUIPMENT*). Era basicamente dessa forma que funcionava o processo de automação dos casos de teste.




```
SELECT 'ETL_ComparisonAdapterQtd' AS TestLinkName,
   CASE
     WHEN count(status)= 0 THEN 'OK' ELSE 'FAIL'
   END AS result

FROM
(
SELECT distinct
   CASE
     WHEN qtdAdapterLegacy = qtdAdapterNew THEN 'OK' ELSE 'FAIL'
   END AS status

FROM
(
    SELECT
        (SELECT count(*)
            FROM  MMT_ADMIN.TBL_DEVICE_INSTANCES@LEQORACL
            WHERE CFK_DEVICENAMES_DEVICETYPE = '1'
        ) AS qtdAdapterLegacy,

        (SELECT count(*)
            FROM T_EQUIPMENT eq
            WHERE eq.TYPE_FK = ( SELECT eqtype.TYPE_PK
                                 FROM T_EQUIPMENT_TYPE eqtype
                                 WHERE eqtype.TYPE_NAME = 'ADAPTER')
        ) AS qtdAdapterNew

    FROM dual
) resultTest

) summaryTest

WHERE status = 'FAIL'
```

**Figura 4 – Exemplo de script completo**

Após todo o planejamento, especificação e automação, as execuções de teste puderam ser feitas. Porém antes de executar cada teste, foi feito um template (parte cabeçalho do teste na Figura 4) de script SQL que ajudava a montar os resultados, ou seja, automatizava a geração de resultados. Este template continha o nome do caso de teste que estava especificado no TestLink. Assim era possível mapear facilmente o caso de teste com a automação correspondente.

Com isso a geração do relatório de execução era automática e exportada do PL/SQL no formato de planilha Excel (Figura 5). Todo este SQL era executado pela ferramenta de ETL *Data Manager*. Esta ferramenta foi configurada para que criasse uma linha na tabela de resultados a cada execução de teste.

| ID | SUMMARY | CATHEGORY | OWNER | ASSIGN |
|---|---|---|---|---|
| 1 | ETL_ComparisonStationType | OK | llobao | ronamart |
| 2 | ETL_ComparisonAdapterQtd | OK | llobao | ronamart |
| 3 | ETL_ComparisonTestBoxQtd | OK | llobao | ronamart |
| 4 | ETL_ComparisonPartsType | OK | | |
| 5 | ETL_ComparisonPartsQtd | FAIL | llobao | ronamart |

**Figura 5 – Exemplo de Relatório Exportado**

Após esta execução e geração de relatório, o cadastro de defeitos era feito através da ferramenta de controle de defeitos. A ferramenta utilizada para cadastro de defeitos foi o Trac [Trac 2010]. Os desenvolvedores recebiam a notificação dos defeitos cadastrados no Trac e trabalhavam na correção.

## 5. Resultados

O processo de teste proposto pôde ser executado em 4 iterações no projeto. Como resultado, observamos que 80% dos defeitos encontrados foram detectados pelas execuções automáticas. Estes eram relacionados aos passos "b" e "d" discutidos na seção 3 deste artigo. Isto ajudou a equipe de desenvolvimento a perceber e consertar defeitos ainda no *Sprint* corrente. Defeitos que eram, em experiências anteriores, encontrados tardiamente, na fase de relatórios, por exemplo, com o processo de teste puderam ser encontrados em um curto espaço de tempo. O custo da resolução destes defeitos, em termos de desenvolvimento, era menor, pois o time já estava trabalhando na funcionalidade.

A maior parte do processo de ETL foi testada. Na migração de dados do legado para o *stage* foi alcançada 100% de cobertura com execuções de teste. Esta cobertura é referente às tabelas de fato e dimensão. Ou seja, todas as tabelas do modelo lógico foram testadas por execuções automáticas. Já os dados do *stage* para o *data mart* obtiveram 80% de cobertura com testes automáticos. Com isso, foi verificado que a acurácia dos dados pôde ser melhorada e que relatórios confiáveis e concisos poderiam ser construídos.

Conforme as iterações iam ocorrendo, e requisitos iam sendo desenvolvidos e alterados, testes de regressão eram executados. Assim defeitos que já tinham sido corrigidos, e que podiam reaparecer, estavam sendo monitorados. De todos os defeitos encontrados, 15% destes eram defeitos recorrentes. A suíte de teste de todo o Projeto continha 136 casos de teste. Destes apenas 3% não foram automatizados. Todos os testes de ETL foram automatizados e executados durante as iterações.

## 6. Conclusão

Com a aplicação deste processo de teste proposto, observamos que foi importante identificar defeitos ainda durante o início do desenvolvimento do Data Warehouse, principalmente na fase de migração dos dados dos sistemas legados, pois o custo desses defeitos quando descobertos tardiamente é muito alto para o projeto, já que demanda um enorme esforço e retrabalho. Um ponto muito importante no processo de teste é que ele permitiu obter a acurácia dos dados legados manipulados, pois com isso há garantia de confiabilidade nas informações geradas.

Nesta experiência, a equipe tomou como lição aprendida que é importante reservar tempo para especificar a modelagem, pois a partir disso podemos encontrar problemas de especificação e formular os testes com antecedência. Além disso, observamos que o elemento chave para apoiar esse processo de teste é a automação dos casos de testes e da geração de relatórios, pois além de economizar tempo com criação de documentos, é possível fazer reuso de código, editar e replicar os testes, o ambiente ajudou para a codificação, execução e elaboração de relatórios ágeis.

Como ações futuras para a melhoria deste processo, a equipe pretende incluir as atividades de teste desde o planejamento do projeto, outra ação é realizar pesquisas para otimizar o ambiente de automação, buscar maneiras de automatizar testes na fase de geração de relatórios e garantir a execução automática de testes de regressão a cada alteração feita no *Data Warehouse*. Com isso, a equipe pretende alcançar a maturidade deste processo de teste e agregar qualidade e valor a esses sistemas que são de importância estratégica para a organização.

## Referências

COLLINS, E. e LOBÃO, L. Experiência em Automação do Processo de Testes em Ambiente Ágil com *SCRUM* e ferramentas OpenSource. IX Simpósio Brasileiro de Qualidade de Software, Relatos de Experiência, 2010.

LEÃO, Leandro e QUAGLIA, Eduardo. Gerenciamento de Escopo em projetos Inovativos Utilizando Metodologias Ágeis. Monografia - Unicamp, Março, 2010.

IBM Cognos Data Manager, "Extract, Transform and Load Framework". http://publib.boulder.ibm.com/infocenter/rentrpt/v1r0m0/index.jsp?topic=/com.ibm.rational.raer.overview.doc/topics/c_datamanager.html, Janeiro, 2010.

KIMBALL, R. and ROSS, M. The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling, John Wiley & Sons, 2th Edition, 2002.

MURPHY, Craig. *Adaptive Project Management Using Scrum*. Methods & Tools, Winter 2004.

Oracle PL/SQL – PL/SQL User's Guide and Reference. http://download.oracle.com/docs/cd/B10500_01/appdev.920/a96624/toc.htm, Janeiro, 2010.

SCHWABER, K. *Agile Project Management with Scrum*. Microsoft Press. 1st edition, 2004.

TESTLINK Testing Tool, "*TestLink Documentation. TEAMST - Home of TestLink developers Community*", http://www.teamst.org/, Novembro, 2009.

TRAC – *Trac User and Administration Guide*. http://trac.edgewall.org/wiki/TracGuide, Janeiro, 2010.