UNIVERSIDADE DO SUL DE SANTA CATARINA

VALDINEI VALMIR DOS SANTOS

DATA WAREHOUSE:
ANÁLISE DA PERFORMANCE DE FERRAMENTAS DE ETL

Florianópolis

2013

VALDINEI VALMIR DOS SANTOS

DATA WAREHOUSE:
ANÁLISE DA PERFORMANCE DE FERRAMENTAS DE ETL

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização da Universidade do Sul de Santa Catarina, como requisito parcial à obtenção do título de Especialista em Engenharia de Projetos de Software.

Orientador: Profº. Aran Bey Tcholakian Morales, Dr.

Florianópolis

2013

VALDINEI VALMIR DOS SANTOS

DATA WAREHOUSE:
ANÁLISE DA PERFORMANCE DE FERRAMENTAS DE ETL

Este Trabalho de Conclusão de Curso foi julgado adequado à obtenção do título de Especialista em Engenharia de Projetos e Softwares e aprovado em sua forma final pelo Curso de Especialização em Engenharia de Projetos de Softwares da Universidade do Sul de Santa Catarina.

Florianópolis, 15 de abril de 2013.

_____

Orientador Prof. Aran Bey Tcholakian Morales, Dr.
Universidade do Sul de Santa Catarina

_____

Profª. Vera R. Niedersberg Schuhmacher, MEng.
Universidade do Sul de Santa Catarina

RESUMO

Este trabalho visa analisar a performance de duas ferramentas de Extração Transformação e Carga (ETL) conceituadas mundialmente no quesito integração de dados. A *Pentaho Data Integration* (PDI), conhecida popularmente como *Kettle*, e a *Talend Open Studio* (TOS). Seguindo os conceitos de *Data Warehouse* (DW), usamos um modelo dimensional no formato estrela, baseado numa arquitetura de rede de telefonia celular, que serviu de base para montagem do processo de ETL analisado. A análise da performance contemplou tanto o tempo necessário para se construir o processo de ETL proposto, como o tempo que este processo leva para ser executado. Com isso obtivemos como conclusão principal que as duas ferramentas avaliadas atendem muito bem qualquer necessidade presente, tanto no processo de ETL proposto como em outros processos mais complexos, e que comparar ferramentas de ETL é uma tarefa muito difícil, pois a mesma necessidade pode ser atendida com a montagem de vários processos de ETL distintos. Sabendo dessa diferença e considerando mesmo assim que processos distintos foram montados em cada ferramenta, o Kettle se mostrou mais eficiente na montagem do processo, e teve uma grande vantagem no tempo de execução deste processo. No ponto de vista do autor a ferramenta Kettle também leva uma pequena vantagem em relação à curva de aprendizagem da ferramenta se comparada à ferramenta TOS.

Palavras-chave: Data Warehouse. Extração Transformação e Carga. Ferramenta. Kettle. Performance. Processo de ETL. Talend Open Studio.

ABSTRACT

This study aims to analyze the performance of two Extraction tools Transformation and Load (ETL) world-renowned in the category of data integration. Pentaho Data Integration (PDI), popularly known as Kettle and Talend Open Studio (TOS). Following the concepts of Data Warehouse (DW), we used a dimensional model in the form star network architecture based on a cell phone, which served as a base for mounting the ETL process analyzed. The analysis of the performance included both the time needed to build the ETL process proposed as the time this takes to run. Thus we have obtained as a conclusion that the two main tools evaluated very well meet any need this, both the ETL process as proposed in other more complex processes, and compare ETL tools is a very difficult task, because the same need can be met with mounting various separate ETL processes. Knowing this difference and considering yet distinct processes that were mounted on each tool, Kettle was more efficient in the assembly process, and had a large advantage in runtime of this process. In point of view author the tool Kettle also leads a small advantage over learning curve of the tool compared to TOS tool.

Keywords: Data Warehouse. Extraction Transformation and Load. Kettle. Performance. Process ETL. Talend Open Studio. Tool.

LISTA DE FIGURAS

LISTA DE QUADROS

SUMÁRIO

1 INTRODUÇÃO

A tecnologia transforma as pessoas, os hábitos e até as profissões, fazendo com que mais profissões surjam e outras sejam extintas. Nessa onda vemos que a área de *Business Intelligence* (BI) vem ganhando espaço nas empresas e contribuindo para essa realidade.

Para que um sistema de BI seja realidade dentro de uma empresa alguns subsistemas precisam ser implantados, e um dos mais importantes nesse contexto é o *Data Warehouse* (DW), pois é ele que faz o trabalho necessário para disponibilizar as informações que o sistema de BI precisa para geração de seus relatórios e suas análises.

No dia a dia de um DW o processo de Extração Transformação e Carga (ETL) é uma constante, e muitas vezes o ponto primordial, pois é ele que alimenta o DW e conseqüentemente o BI. Se esse processo não funcionar de forma adequada o BI não terá as informações necessárias.

Diante da importância do processo de ETL algumas ferramentas foram criadas para facilitar sua construção e manutenção, que por alguns anos foi um processo moroso e de difícil manutenção.

1.1 PROBLEMA

Considerando a altíssima relevância do processo de ETL no contexto de um DW, vemos muitos projetos usando ferramentas de ETL distintas e posições pessoais vagas sobre qual ferramenta se enquadra melhor a um determinado projeto.

Com isso, vemos a necessidade de conhecer os benefícios e dificuldades de algumas ferramentas de ETL e avaliar qual seria a mais adequada para esse processo.

1.2 JUSTIFICATIVA

Com o passar dos anos muitas ferramentas de ETL surgiram no mercado, e muitas já adquiriram uma maturidade considerável, levando as mesmas a serem usadas por grandes corporações e a ser objeto de desejo de outras. Muitas vezes a decisão de usar uma ferramenta de ETL específica pode nos trazer problemas futuros, sendo esse processo de escolha uma tarefa árdua quando não temos subsídios ou informações suficientes para fazermos a melhor escolha.

Com base no exposto, vemos que um *Benchmark* entre algumas das ferramentas de ETL disponíveis no mercado, nos traria uma base sólida, que serviria de apoio à tomada de decisão sobre qual a ferramenta de ETL se adéqua melhor a determinado projeto. Além de nos fornecer uma descrição sobre cada ferramenta e assim contribuir para o aprendizado das mesmas.

1.3 OBJETIVOS

1.3.1 Objetivo Geral

Considerando o contexto de um *Data Warehouse*, este trabalho tem como objetivo analisar e comparar duas ferramentas de ETL.

1.3.2 Objetivos Específicos

- Analisar tempo de execução do processo de ETL em cada ferramenta, com base no cenário proposto;
- Analisar a facilidade em montar o cenário proposto na respectiva ferramenta;
- Analisar a facilidade em diagnosticar problemas no cenário proposto;
- Conhecer novas ferramentas de ETL.

2 BUSINESS INTELLIGENCE – BI

De acordo com INMON (1997), os Sistemas de Apoio à Decisão (SAD), que atualmente chamamos de BI, passaram a ser uma realidade somente a partir da década de 1980, e tiveram uma longa e complexa evolução até a sua maturidade. Essa evolução começou na década de 1960 e só foi possível graças ao surgimento de algumas tecnologias, como o armazenamento em disco (*Direct Access Storage Device* - DASD) e os Sistemas Gerenciadores de Banco de Dados (SGBD), no início da década de 1970, os PCs (*personal computer*) e as linguagens de 4ª geração (L4G), antes do início da década de 1980, e o programa de extração por volta de 1985.

2.1 CONCEITO

Segundo Sezões, Oliveira e Baptista, *Business Intelligence* é:

> Conceito que engloba um vasto conjunto de aplicações de apoio à tomada de decisão que possibilitam um acesso rápido, partilhado e interativo das informações, bem como a sua análise e manipulação; através destas ferramentas, os utilizadores podem descobrir relações e tendências e transformar grandes quantidades de informação em conhecimento útil. (SEZÕES, OLIVEIRA e BAPTISTA, 2006, p.10).

Os autores ainda expõem que BI refere-se à simbiose entre a gestão e tecnologia. Deixando claro que somente a tecnologia não irá sustentar um sistema de BI, e que a gestão tem papel importante para o sucesso deste tipo de sistema.

Também é exposto que BI é “um processo produtivo cuja matéria-prima é a informação e o produto final o conhecimento”, resumindo de forma sucinta o foco principal de um sistema de BI é transformar a grande quantidade de informação que as organizações têm disponível em informações realmente úteis, ou mais precisamente em conhecimento sobre suas próprias atividades e negócios.

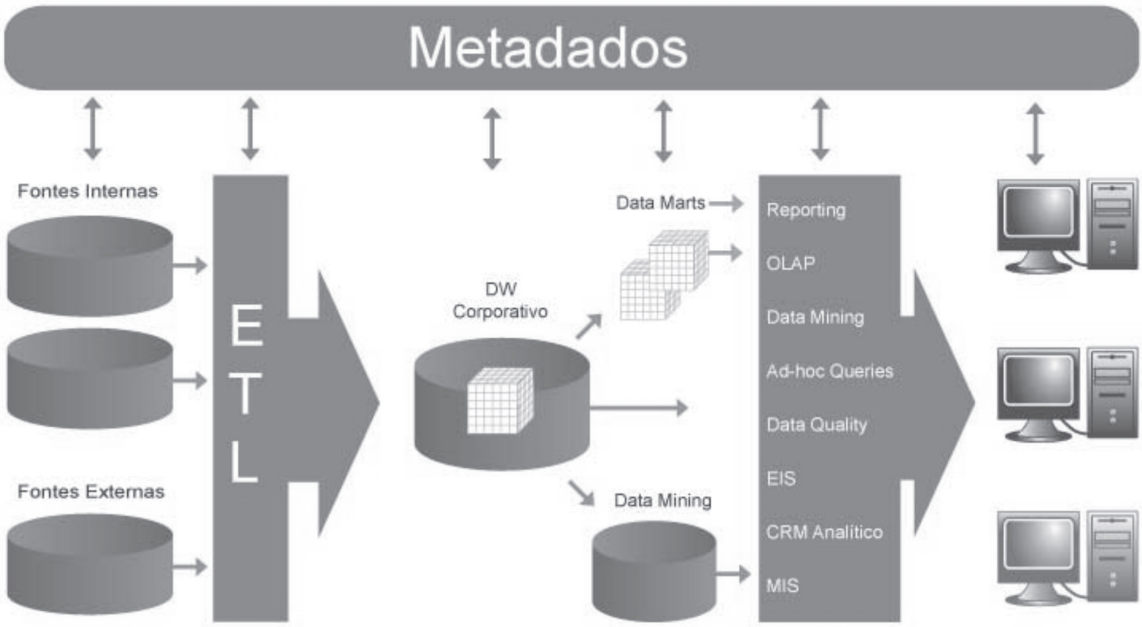
2.2 ARQUITETURA

Para Sezões, Oliveira e Baptista (2006), um sistema de BI tecnologicamente se enquadra como um sistema de informação da organização, o qual não existe sozinho, pois está ligado de forma umbilical às fontes de dados.

Na arquitetura tecnológica de um BI, exposta na a Figura 1, fica evidente essa dependência das fontes de dados, que fazem parte da base do sistema.

Nessa figura ainda podemos observar a existência do processo de ETL, que disponibiliza as informações para povoamento do DW e *Data Marts* (DM). Informações estas que posteriormente serão analisadas pelas aplicações de *Front-end*, que são a parte visível ao usuário num sistema de BI.

Figura 1 – Estrutura típica e simplificada de um sistema tecnológica de Business Intelligence
Fonte: Sezões, Oliveira e Baptista, 2006



3 DATA WAREHOUSE - DW

3.1 CONCEITO

Existem diversas definições sobre o que é um *Data Warehouse* (DW), dentre elas temos a de Inmon (1997, p.33), que se caracteriza como um dos principais autores nesta área. Ele reporta o seguinte: “um data warehouse é um conjunto de dados *baseado em assuntos*, *integrado*, *não-volátil*, e *variável em relação ao tempo*, de apoio às decisões gerências.”

Diante do conceito apresentado, fica explícito que o foco principal de um DW está no apoio às decisões gerenciais, que são tomadas através da análise do conjunto de dados fornecido pelo DW.

Para Barbieri (2001, p.49), podemos definir DW como um Banco de Dados destinado a sistemas de suporte ao apoio a decisões, cujos dados armazenados seguem o modelo dimensional, que possibilita o processamento analítico pelas ferramentas de *On-Line Analytical Processing* (OLAP) e *Mining*.

Sezões, Oliveira e Baptista (2006), acrescentam que um DW é um repositório de informação, que agrupa os diversos dados históricos, facilitando as tarefas de análise e reporting de uma organização, e que muitas vezes estas informações precisam ser divididas em conjuntos menores e agrupadas de forma lógica em pequenas unidades, que chamamos de *Data Marts* (DM).

Os mesmos autores comentam que a criação de um DW justifica-se por dois motivos fundamentais:

- Para a integração de dados distribuídos em diferentes fontes de dados, tendo em vista a necessidade de uma análise global;

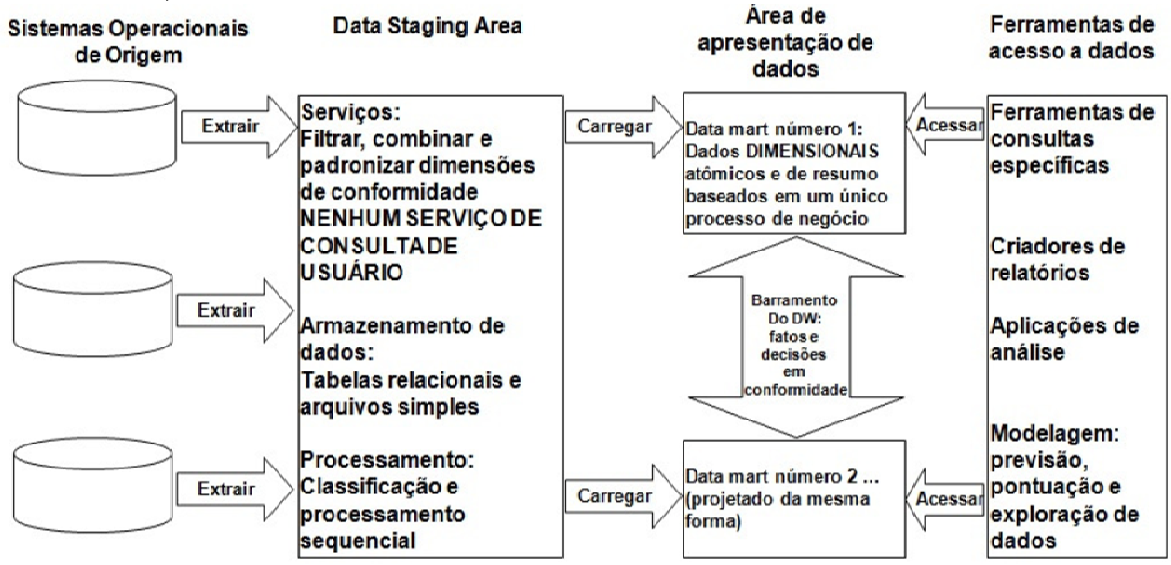
- Para separação dos dados operacionais dos dados usados para análise e reporting, tendo em vista a tomada de decisão.

3.2 ARQUITETURA

Para Kimball (2002) um DW está dividido em quatro componentes distintos. Conforme a Figura 2, estes componentes são: Sistemas operacionais de origem (fontes de dados), data staging área, área de apresentação de dados e ferramentas de acesso a dados.

Estes quatro componentes resumem o que é um DW, e por este motivo os veremos com mais detalhes nos próximos tópicos.

Figura 2 – Elementos básicos do data warehouse
Fonte: Kimball, 2002



3.2.1 Sistemas operacionais de origem

São os sistemas transacionais existentes nas empresas. Esses sistemas guardam os dados operacionais da empresa, e servem de fonte de dados para o DW.

Para Kimball (2002), eles devem ser considerados como externos ao DW, porque geralmente temos pouco ou nenhum controle sobre o conteúdo e o formato dos dados nesses sistemas operacionais. Eles têm como prioridade principal o desempenho e a disponibilidade de processamento, e geralmente matem um volume

pequeno de dados históricos. Também é comum que cada sistema de origem seja uma aplicação independente e não tenha nenhum compartilhamento de dados comuns entre elas.

3.2.2 Data staging area

É uma área temporária usada pelo DW para fazer o processo de ETL. Tendo como foco principal a extração dos dados de diversas fontes, a transformação desses dados conforme a necessidade, e a carga deles na base de dados dimensional usada pelo DW.

Conforme Kimball (2002), essa área é representada por tudo que existe entre os sistemas operacionais de origem e a área de apresentação de dados. O autor ainda faz uma bela comparação entre a *Data Staging Area* e a cozinha de um restaurante. Na cozinha os alimentos crus são transformados em refeições. Já no DW os dados operacionais brutos são transformados em um formato usado pelo DW e ficam prontos para serem consultados (consumidos). Tanto a cozinha do restaurante como a *Data Staging Area*, só podem ser acessadas por profissionais qualificados e nem tudo que acontece em ambos pode ser observado pelos clientes.

O requisito principal da arquitetura de uma *Data Staging Area* é que ela não seja acessada pelos usuários do DW e que não forneça serviços de consulta ou apresentação de dados.

3.2.3 Área de apresentação dos dados

A área de apresentação dos dados é o DW propriamente dito. É aqui que ficam todos os dados disponíveis para o acesso dos usuários e de outras aplicações. Como a staging área é inacessível, está área funciona como o DW para a comunidade de negócios da empresa.

Normalmente a área de apresentação está estruturada pela composição de vários *Data Marts* (DM) integrados. Cada um deles representando os dados de um único processo de negócio da empresa.

Na grande maioria dos DW vemos a área de apresentação disponível em um Banco de Dados relacional, e seguindo um modelo de dados específico para o uso em DW, que é o "Modelo Dimensional".

Segundo Kimball (2002), a área de apresentação também pode se basear em tecnologias de banco de dados multidimensional ou de processamento analítico on-line (OLAP), que armazenam seus dados em "cubos"[1].

A maioria dos "cubos" OLAP tem origem em modelos dimensional presentes em bancos de dados relacionais, seguindo o "esquema em estrela", que é o esquema mais usado na concepção de um DW.

3.2.3.1 Modelo Dimensional

Este modelo é difundido mundialmente e pela sua grande aceitação comprova-se que é a técnica mais viável para acesso aos dados de um DW, pois torna os bancos de dados mais fáceis e compreensíveis.

3.2.3.1.1 Esquema em estrela (start schema)

Sezões, Oliveira e Baptista (2006, p.36), definem o *esquema em estrela* como:

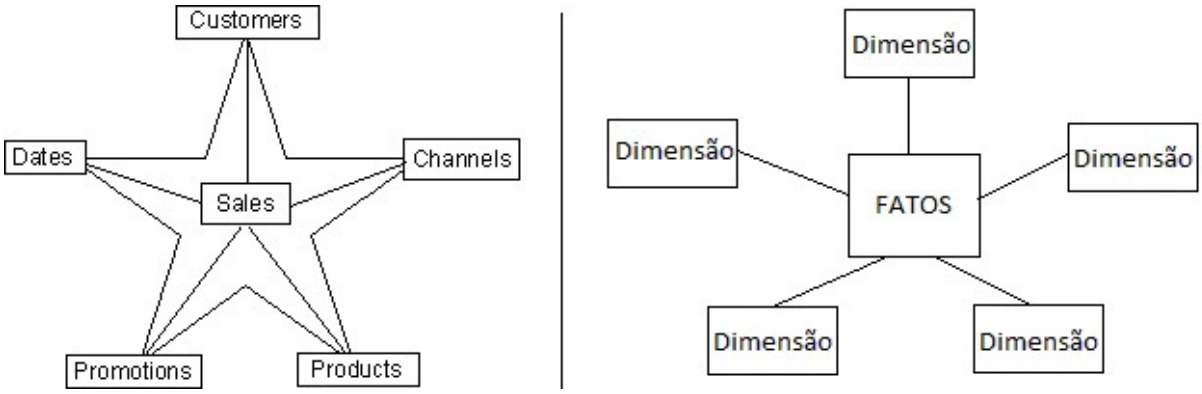
> a criação de tabelas dimensionais (e.g., dimensão tempo, dimensão clientes, dimensão geográfica), que ficam ligadas entre si através de uma tabela de fatos. A sua interligação baseia-se num esquema lógico e simples: as tabelas dimensionais contêm as definições das características dos eventos, enquanto as tabelas de fatos, por sua vez, armazenam os fatos decorridos e as chaves estrangeiras para as características respectivas que se encontram nas tabelas dimensionais.

[1] - Uma estrutura dimensional em uma plataforma de banco de dados OLAP ou multidimensional, inicialmente referindo-se a um caso simples de três dimensões: produto, mercado e hora, por exemplo (Kimball, 2002).

> Este modelo apresenta vantagens óbvias, como por exemplo, a existência de uma única tabela de fatos contendo toda a informação sem redundâncias, a definição de apenas uma chave primária por dimensão, a redução do número de interligações e a conseqüente pouca necessidade de manutenção.

A designação de esquema em estrela vem do simples fato da estrutura física do modelo de dados ser parecida com a estrutura de uma estrela, conforme visto na comparação feita na Figura 3.

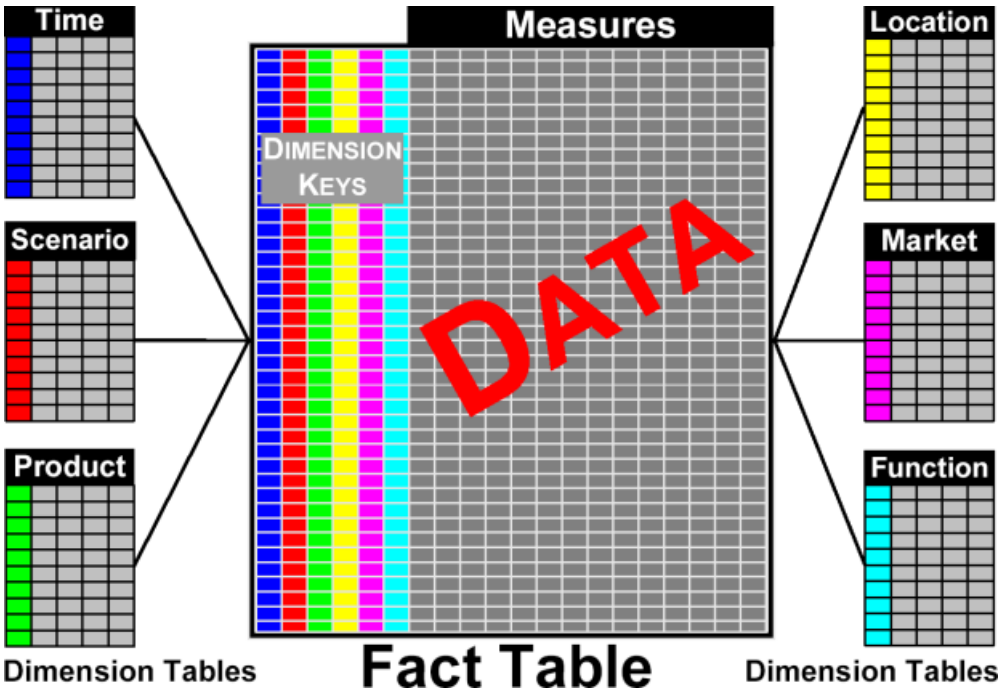
Figura 3 – Exemplo modelo dimensional com esquema em estrela
Fonte: Technet, adaptado pelo autor, 2012



3.2.3.3 Tabela de fatos

Para Kimball (2002, p. 21), na modelagem dimensional temos a tabela de fatos como à principal tabela do modelo. É nela que todas as medições numéricas ficarão armazenadas, como visto no exemplo da Figura 4. A palavra “fato” é usada para representar uma medição de negócio da empresa, ou seja, cada linha da tabela representa uma medição, e todas essas medições devem estar alinhadas na mesma “granularidade”.

Figura 4 – Relacionamento entre tabela fato e tabelas dimensão
Fonte: The BI Verdict, 2012



Os fatos mais úteis são numéricos e aditivos, por exemplo, o volume de vendas em reais, que é um valor numérico, possibilita a adição dele com outros valores de volume de venda, podendo assim, trazer a informação de outros períodos.

3.2.3.4 Granularidade

Segundo Inmon (1997, p.45), é o aspecto mais importante em um projeto de DW, pois ela é responsável pelo nível de detalhe contido nas unidades de dados existentes no DW. O autor também expõe que “Quanto mais detalhe, mais baixo o nível de granularidade. Quanto menos detalhe, mais alto o nível de granularidade.”

A granularidade sempre é uma questão delicada num projeto de DW, e essa preocupação se dá pelo fato dela afetar profundamente o volume de dados armazenados pelo DW, e ao mesmo tempo afetar o tipo de consulta que iremos conseguir extrair do DW.

3.2.3.5 Tabelas de dimensão

Já em relação às tabelas de dimensão, Kimball (2002, p. 24), reporta que elas sempre estarão acompanhadas de uma tabela de fatos, e conterão descrições textuais. Como exemplo, podemos citar uma tabela dimensão produto, em que as colunas da tabela seriam a descrição do produto, a descrição da categoria, a descrição do peso, e outras descrições inerentes ao produto. As tabelas de dimensão geralmente são tabelas simples em relação ao número de linhas, mas podem conter um número muito grande de colunas. Cada dimensão tem sua definição com base em sua chave primária, e sua ligação com a tabela de fatos fica bem compreendida com o auxilio da Figura 4.

Os atributos (colunas) das tabelas dimensão têm um papel importantíssimo num DW, pois são a origem para os rótulos usados nos relatórios e são fundamentais para fazer com que o DW possa ser usado e compreendido. Para Kimball (2002, p.24), “sob muitos aspectos, a utilidade do data warehouse depende dos atributos de dimensões. A potência do data warehouse é diretamente proporcional à qualidade e à complexidade dos atributos de dimensões.” O autor ainda expõe que quanto mais nos dedicarmos à inclusão de atributos bem descritivos, melhor será o DW.

Os melhores atributos são os textuais e os discretos, formados por palavras reais e não abreviações.

3.2.4 Ferramentas de acesso a dados

São todas as aplicações que acessam a área de apresentação dos dados. Estas aplicações podem ser simples e de consultas específicas, ou complexas e sofisticadas para exploração de dados.

3.2.5 Metadados

A arquitetura de um DW não estaria completa se deixássemos os metadados de lado, pois onde iríamos guardar as informações sobre o processo de transformação feito na extração e carga de uma respectiva tabela, por exemplo, ou as informações sobre o período em que a carga dessa tabela será feita, e assim por diante. É impossível termos um DW sem nos preocuparmos em guardar essas informações.

Podemos dizer que os metadados se referem a todas as informações geradas no DW, que não sejam os dados propriamente ditos. Os metadados são considerados por Kimball (2002, p.18), como uma enciclopédia para o DW, pois neles estarão catalogadas todas as informações do DW, da mesma forma que são catalogados os recursos de uma biblioteca.

4 EXTRAÇÃO TRANSFORMAÇÃO E CARGA – ETL

A sigla ETL traduz muito bem o papel dessa fase na arquitetura de um BI. É nessa fase que os dados são extraídos dos ambientes operacionais e levados a uma área temporária para serem transformados (limpos, agregados etc.), e posteriormente carregados na base de dados do DW, ficando disponíveis para as aplicações de análise de dados do BI.

Conforme Inmon (1997), a fase de ETL muitas vezes é vista como simples por alguns programadores, fazendo com que estes imaginem ETL como uma simples extração de dados do ambiente operacional para a carga no ambiente do DW. Mas o que parece simples na verdade é a fase mais importante e a mais complexa na implantação de um BI, pois selecionar os dados relevantes de várias bases heterogêneas e trazê-los para o padrão definido no DW é uma tarefa onerosa.

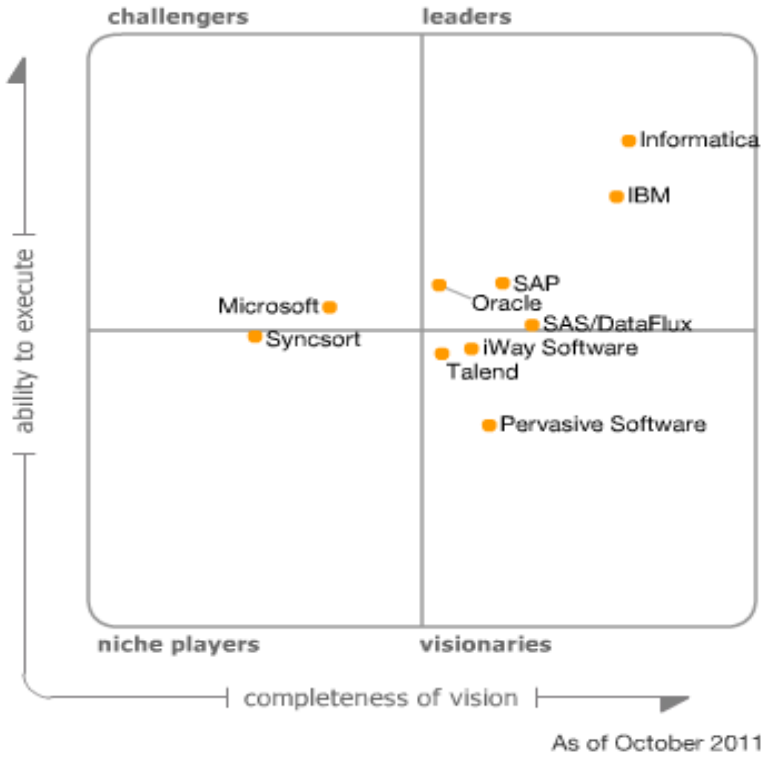
Na visão de Sezões, Oliveira e Baptista (2006, p.51), “ETL é um conjunto de processos que permite às organizações extrair dados de fontes de informação diversas e reformulá-los e carregá-los para uma nova aplicação (base de dados, geralmente um data warehouse) para análise”.

5 FERRAMENTAS DE ETL

Conforme Barbieri (2001, p.72), as ferramentas de ETL têm o objetivo de realizar todo o processo de ETL, sendo estas geralmente desenvolvidas internamente pelas empresas, ou adquiridas no mercado. A qualidade dos dados em um DW é extremamente importante, e a escolha da ferramenta certa, junto à integridade das fontes de dados, são fatores críticos para o sucesso de qualquer projeto de DW.

Existem diversas ferramentas disponíveis no mercado atualmente, e todas atendem de forma satisfatória vários tipos de transformações que o processo de ETL necessita. As ferramentas mais conceituadas são de empresas consideras gigantes no mercado de Tecnologia da Informação (TI), como a IBM e a Oracle, por exemplo, mas devido ao alto custo dessas ferramentas várias empresas acabam optando por soluções open source.

Figura 5 – Magic Quadrant for Data Integration Tools
Fonte: Gartner Group, 2011



Na Figura 5 podemos observar o quadrante mágico da Gartner Group (empresa de consultoria conceituada mundialmente), o qual tem a finalidade de avaliar os fornecedores dentro de um mercado especifico. O quadrante desta figura avalia os fornecedores de ferramentas de integração de dados conforme sua atuação neste mercado. No quadrante temos a classificação dos fornecedores como Lideres (Leaders), Desafiadores (Challengers), Visionários (Visionaries) ou como fornecedores com foco no nicho de mercado (Niche Players). Nesta figura fica visível a disputa existente neste mercado e como estava à classificação desses fornecedores em outubro de 2011.

A Gartner Group ainda expõe que nem sempre um fornecedor classificado como líder de mercado é a melhor escolha, pois um fornecedor classificado como desafiador ou um com o foco no nicho de mercado podem atender muito bem as necessidades, as quais vão depender dos seus objetivos de negócio. Já os fornecedores visionários estão muito perto dos líderes e se continuarem nesse caminho logo serão líderes, mostrando assim que também podem ser uma boa escolha.

Das ferramentas open source disponíveis no mercado, temos duas que estão mais presentes nas empresas brasileiras, que são elas a *Talend Open Studio* (TOS) e a *Pentaho Data Integration* (PDI), esta conhecida como *Kettle*. Destas ferramentas somente a *TOS* aparece no quadrante mágico da Gartner Group, ficando evidente seu destaque sobre a ferramenta *Kettle*.

5.1 KETTLE

Figura 6 – Logo oficial da suíte Pentaho
Fonte: Imagem disponível junto com ferramenta, 2012

Segundo o site oficial da ferramenta Pentaho Data Integration (PDI), conhecida como Kettle, ela proporciona uma poderosa extração, transformação e

carregamento (ETL) usando uma inovadora abordagem orientada a metadados. Com uma interface gráfica intuitiva, num ambiente que usa drag and drop (arrasta e solta) dos componentes, e uma comprovada arquitetura baseada em padrões e escalável. O Kettle é cada vez mais a escolha para as organizações mais tradicionais, que buscam uma ferramenta específica de ETL ou ferramentas para integração de dados.

A edição comunitária da ferramenta é um software de código aberto e auto suportado. Já a versão Enterprise Edition (EE) inclui suporte técnico, atualizações e recursos corporativos, mas não é gratuita.

5.2 TALEND OPEN STUDIO

Figura 7 – Logo oficial da suíte Talend Open Studio
Fonte: Talend, 2012

No site da empresa *Talend*, é exposto que os produtos de integração de dados, fornecidos por eles, proporcionam uma integração poderosa e flexível, de modo que as empresas não precisam mais se preocupar em saber como as bases de dados e as aplicações estão conversando entre si, e sim se preocupar em maximizar o valor do uso de dados.

Também é exposto que a ferramenta é extensível, altamente escalável, de código aberto e com um conjunto de ferramentas para acessar, transformar e integrar dados de qualquer sistema de negócios em tempo real ou em lote para atender tanto as necessidades operacionais e analíticas de integração de dados. Com mais de 450 componentes de conexão, ela se conecta a praticamente qualquer fonte de dados. A ampla gama de casos de uso da ferramenta Talend incluem: ETL para *Business Intelligence* (BI), *Master Data Management* (MDM) e *Data Warehouse* (DW), sincronização de dados, migração de dados e consolidação; compartilhamento de dados e serviços de dados.

6 DESENVOLVIMENTO

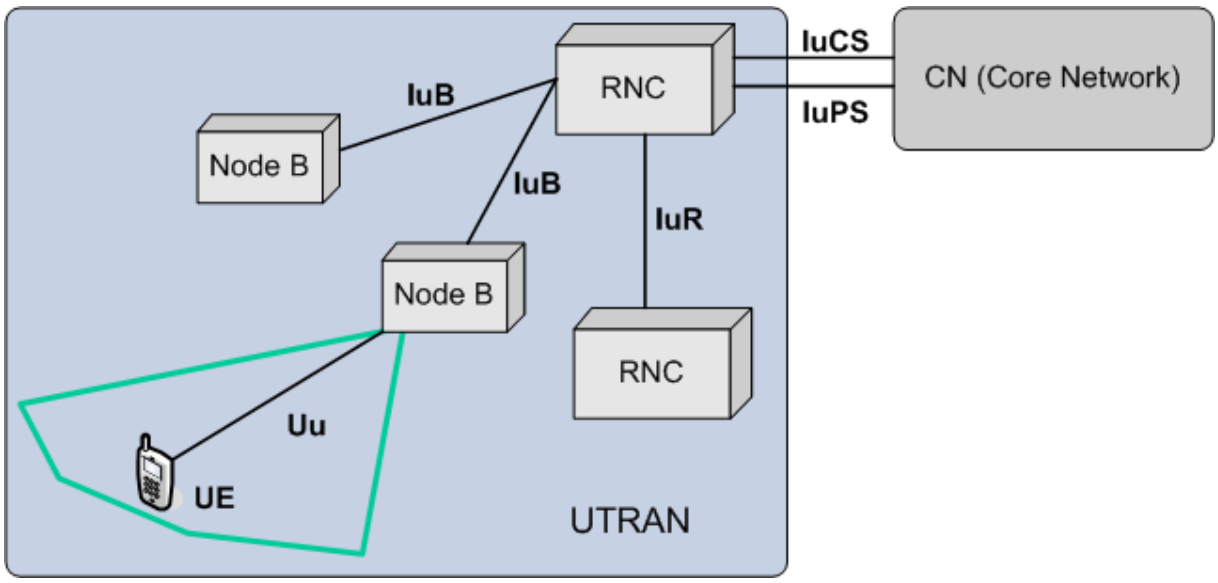
Este trabalho de conclusão de curso tem como objetivo geral analisar e comparar duas ferramentas de ETL, pois atualmente temos muitas ferramentas no mercado e certa dificuldade em saber qual se enquadra melhor a determinado projeto de DW. Essa análise nos dará um alicerce para a tomada de decisão e também um detalhamento sobre algumas funcionalidades e facilidades dessas ferramentas.

6.1 ESTRUTURA USADA COMO BASE

A estrutura escolhida para servir de base para a análise das ferramentas de ETL vem do ramo das telecomunicações, e tem como base a coleta de dados de contadores específicos de alguns elementos que compõem uma rede de telefonia celular 3G (terceira geração). A arquitetura exposta na Figura 8 mostra como estes elementos estão distribuídos em parte dessa rede, que é dividida em UTRAN (*UMTS Terrestrial Radio Access Network*, sendo UMTS a sigla para *Universal Mobile Telecommunications System*) e CORE (núcleo da rede). Os elementos que estaremos recebendo dados estão na área UTRAN e serão as RNC (*Radio Network Controller*) e as Node-B (termo usado para interface aérea de comunicação com as unidades móvel, também conhecida como *estações base*).

Na Figura 8 podemos notar que existe uma hierarquia entre esses elementos, onde toda Node-B sempre estará vinculada a uma RNC apenas e poderá ter vários UE (*User Equipment*) conectados a ela. As RNC além de controlar as Node-B podem estar conectadas a outras RNC e a outro elemento de maior hierarquia na área CORE.

Figura 8 – Arquitetura da área UTRAN
Fonte: Wikipédia, 2012



6.2 ARQUITETURA DO DW

Levando em consideração a arquitetura de um DW, definida por Kimball (2002), e o foco na análise das ferramentas de ETL, somente os componentes *Sistemas operacionais de origem (fontes de dados), data staging área e Área de apresentação dos dados* serão necessários para efetuarmos a análise das ferramentas de ETL. O componente *Ferramentas de acesso a dados* não será abordado nesta análise.

6.2.1 Sistemas operacionais de origem

O sistema operacional de origem (fonte de dados) de nossa análise será um arquivo texto, em formato CSV (*Comma-separated values*), fornecido pelo fabricante da rede celular 3G, com uma amostra de dados de um intervalo de três horas. Essa amostra faz parte de um processo de ETL usado pela empresa que fornece a solução de DW a uma operadora de telefonia celular do Brasil. O arquivo supracitado possui 5139 linhas e será submetido às respectivas ferramentas de ETL

analisadas para fazermos as transformações necessárias e a carga na base de dados do DW.

6.2.2 Data staging area

Considerando o uso das ferramentas de ETL a *data staging area* estará intrínseca a cada uma, ou seja, cada ferramenta tem sua forma particular de tratar os dados vindos da fonte de dados. Algumas ferramentas usam o auxilio de SGBDs para efetuar o processo de ETL, outras usam apenas seus próprios recursos, como a memória RAM ou o hard disk do servidor onde esta sendo executada a aplicação.

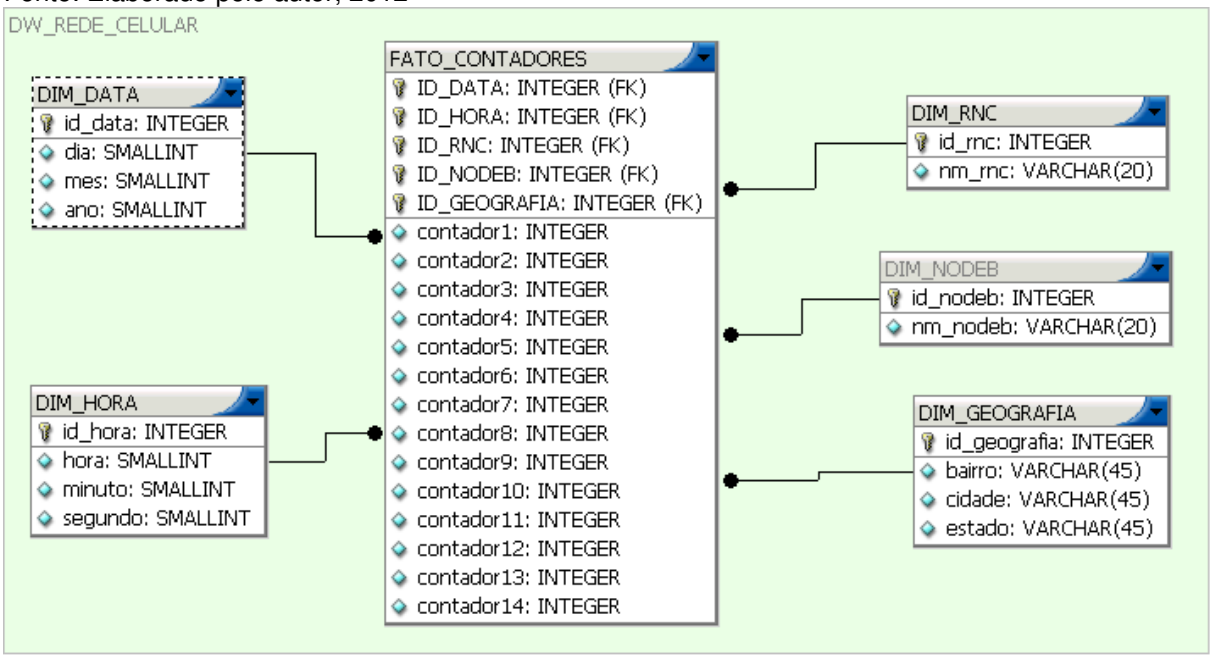
6.2.3 Área de apresentação de dados

Nesta área serão armazenados os dados resultantes do processo de ETL. Aqui os dados já estarão tratados e adequados às arquiteturas e padrões definidos para um DW, conforme veremos nos itens a seguir.

6.2.3.1 Modelo dimensional usado

Para fazer a análise das ferramentas de ETL, à estrutura do modelo dimensional exposta na Figura 9, que segue o modelo estrela, foi montada com base na rede de telefonia celular 3G e nos dados fornecidos pelo arquivo CSV que servirá como fonte de dados.

Figura 9 – Modelo Dimensional do DW proposto
Fonte: Elaborado pelo autor, 2012



6.2.3.2 Tabela de fatos usada

No modelo dimensional estrela essa tabela é responsável por todas as medições de um DW. No nosso DW especificamente, as medições se resumem nos contadores extraídos dos elementos Node-B, da rede de telefonia celular 3G analisada.

A tabela de fatos está definida na Figura 9, com o nome de FATO_CONTADORES e é responsável por guardar os dados de 14 contadores, fornecidos no arquivo CSV disponibilizado como fonte de dados. O nome desses contadores foi alterado para “contadorN”, onde o N foi usado como um número seqüencial, pois para nossa análise o nome dos contadores é irrelevante e dessa forma também não estaremos expondo dados sem autorização.

6.2.3.3 Granularidade usada

Considerando o arquivo de dados fornecido como fonte de dados, a menor granularidade possível já está limitada pelo arquivo, pois não existe a possibilidade de chegarmos a um nível maior de detalhamento se não temos dados menores dessa dimensão disponível. Já um nível maior de granularidade poderemos obter com o agrupamento dos dados.

Diante disso iremos usar a menor granularidade possível, levando em conta que conseguiremos um maior detalhamento dos dados e desconsiderando o fato de termos um maior volume de dados armazenados.

6.2.3.4 Tabelas de dimensão usadas

Num modelo dimensional estrela, as tabelas de dimensão sempre estarão acompanhadas de uma tabela FATO, e serão responsáveis pelas descrições textuais que temos em nosso DW.

Conforme o modelo dimensional da Figura 9, as dimensões usadas em nosso DW serão:

- DIM_DATA, que é a dimensão usada para guardar uma data específica e é representada pelos atributos DIA, MÊS e ANO;

- DIM_HORA, que é a dimensão usada para guardar às 24 horas possíveis de um dia, e é representada pelos atributos HORA, MINUTO e SEGUNDO;

- DIM_RNC, que é a dimensão usada para guardar informações de uma RNC específica, tendo como único dado relevante o nome da RNC, que é representado pelo atributo NM_RNC;

- DIM_NODEB, que é a dimensão usada para guardar informações de uma NODE-B específica, tendo como único dado relevante o nome da NODEB, que é representado pelo atributo NM_NODEB;

- DIM_GEOGRAFIA, que é a dimensão usada para guardar as informações da localização de uma RNC ou NODE-B específica.

6.3 ANÁLISE DAS FERRAMENTAS DE ETL

Para fazermos a análise das ferramentas de ETL usaremos um SGBD Relacional MySQL, para servir de “área de apresentação de dados” do nosso DW. Neste SGBD iremos montar duas bases de dados distintas, uma base para cada ferramenta de ETL avaliada, seguindo o nosso modelo dimensional estrela. As bases terão os nomes dw_kettle e dw_talend. Cada ferramenta de ETL será responsável por “extrair” os dados da fonte de dados, que se resume no arquivo CSV, que estará disponibilizado num diretório padrão para todas as ferramentas analisadas, efetuarem as “transformações” necessárias e a “carga” na base de dados específica da ferramenta analisada. A transformação que será feita consiste em extrair do arquivo texto as colunas necessárias para povoar cada tabela destino e fazer os ajustes necessários nos dados, além de avaliar se tal informação realmente é necessária e não será duplicada na base de dados do nosso DW.

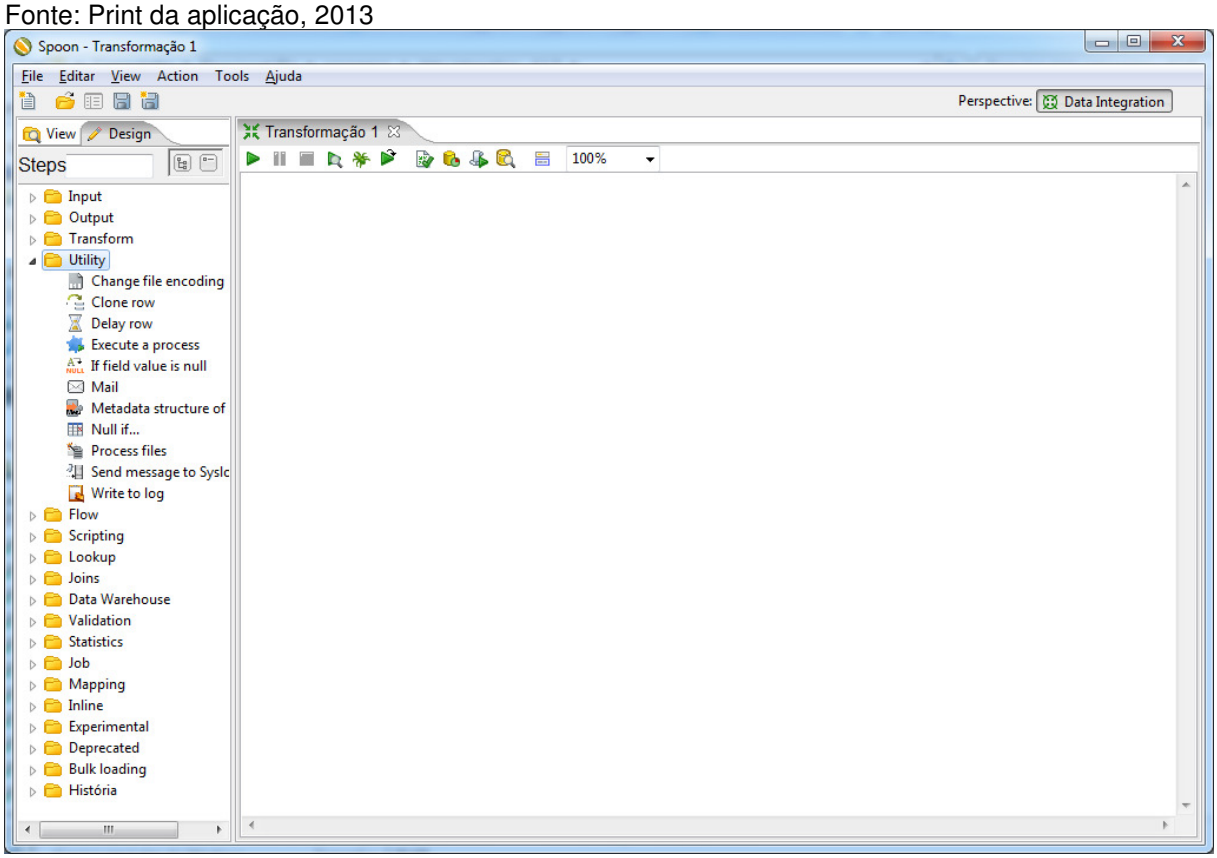
Diante desse cenário será possível avaliar as dificuldades e facilidades encontradas no uso de cada ferramenta, para montar o processo de ETL. Lembrando é claro, que cada usuário possui conhecimentos distintos sobre cada ferramenta, o que os levará a terem dificuldades e facilidades diferentes na hora de montar um processo de ETL. Também será avaliado o tempo levado por cada ferramenta para executar o processo. Lembrando também, que o tempo decorrido para execução de cada processo, depende muito de como este processo foi construído dentro das facilidades disponíveis de cada ferramenta, o que nos leva a crer que os processos montados não necessariamente serão totalmente compatíveis em nível de comparação do tempo decorrido.

Com isso teremos alguns dados relevantes sobre cada ferramenta, o que nos auxiliará na decisão do uso de certa ferramenta para popular futuros DWs.

As ferramentas avaliadas serão a *Pentaho Data Integration* (PDI), mais conhecida no mercado como *Kettle*, e a *Talend Open Studio* (TOS). Estas ferramentas foram escolhidas pelo simples fato de ser *open source*, o que nos isenta de qualquer problema com pirataria de softwares e viabiliza o uso dessas ferramentas por qualquer empresa de baixo poder aquisitivo.

6.3.1 Kettle

Figura 10 – Interface da aplicação Spoon, da ferramenta Kettle, versão 4.1.0
Fonte: Print da aplicação, 2013



A ferramenta Kettle, que possui a interface gráfica, exposta na Figura 10, é escrita em Java e necessita da JVM (*Java Virtual Machine*) para executar qualquer uma das suas três aplicações principais, as quais executam tarefas distintas dentro da ferramenta.

Temos a aplicação *Spoon*, que é a interface gráfica responsável pela montagem do processo de ETL. Processo esse que nesta ferramenta está dividido entre "transformações" e "jobs". Tanto as Transformações como os jobs são compostos por "*steps*" (passos), que são ligados uns aos outros através de um "*hop connection*", que visualmente é representado por uma linha com uma seta indicando sua direção, e tem a função de transferir o fluxo de dados do step de origem para o step destino. Nas transformações os steps representam a menor unidade do processo de ETL, e nos Jobs representam alguma tarefa a nível gerencial do processo. Como exemplo, podemos citar o *step* "*CSV file input*", que é responsável

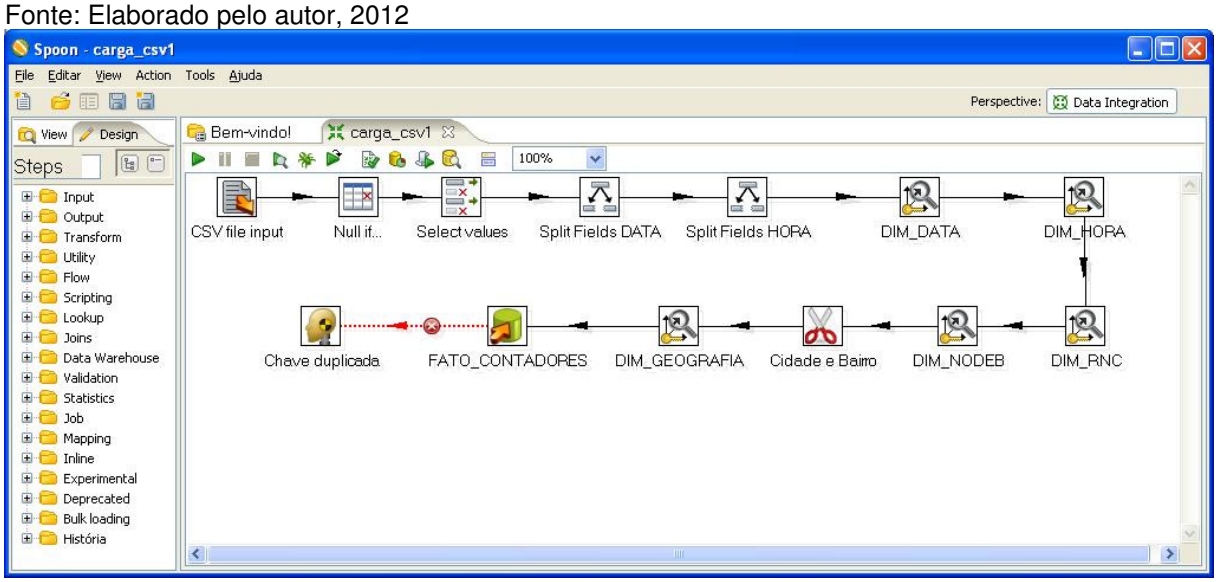
pela entrada de dados de um arquivo CSV, e está disponível apenas para o uso em transformações. Já o step "*Mail*", que é usado para enviar um e-mail, está disponível apenas para o uso em jobs. Essa definição muitas vezes foge a essa regra e podemos encontrar steps em nível de Jobs que executem tarefas que estão fora do contexto gerencial. Os dois principais steps em nível de jobs são o "*Transformation*" e o "*Job*", que permitem executar uma transformação específica e executar um *job* específico, respectivamente. Com isso, podemos criar um *job* principal que seja responsável por executar várias transformações ou jobs encadeadas. Tanto os jobs como as transformações geram arquivos em formato XML como resultado do processo, sendo possível a manipulação direta destes, mas não aconselhada, pois podemos gerar alguma inconsistência e inviabilizar sua edição futura na aplicação *Spoon*. Tanto as transformações como os jobs podem ser executados diretamente na aplicação, e possuem vários recursos para testes e visualização dos dados trafegados no fluxo.

Temos a aplicação *Kitchen*, que é responsável pela execução dos jobs montados pela aplicação *Spoon*, e esses jobs podem estar em formato XML, ou armazenados num repositório. A principal funcionalidade dos jobs é a possibilidade de agendar a execução dos mesmos para qualquer dia ou hora, conforme a necessidade.

Temos a aplicação *Pan*, que é responsável pela execução das transformações montadas pela aplicação *Spoon*, e essas transformações podem estar em formato XML, ou armazenadas num repositório.

6.3.1.1 Montagem do processo de ETL

Figura 11 – Processo ETL montado no Kettle
Fonte: Elaborado pelo autor, 2012



Para montar o processo de ETL proposto na ferramenta Kettle, desenvolvemos apenas a *transformação* exposta na Figura 11, na qual foram necessários 13 steps para conclusão de todo o processo. Estes steps estão detalhados abaixo:

1º - CSV file input: Step responsável pela extração dos dados do arquivo CSV;

2º - Null if: Step responsável pelo tratamento dos campos que chegam com valor NULL. Tendo esses campos o valor NULL alterado para zero, para evitar erros de conversão de tipos, que ocorrem mais pra frente no processo de ETL;

3º - Select Values: Step responsável pela conversão de tipos de dados dos campos vindos do arquivo e exclusão de campos indesejados;

4º - Split Fields Data: Step responsável por criar três novos campos (dia, mês e ano) com base no campo data;

5º - Split Fields Hora: Step responsável por criar dois novos campos (hora e minuto) com base no campo hora;

6º - DIM_DATA: Step do tipo "*Combinação Lookup / Update*", responsável por inserir os dados nos campos dia, mês e ano na tabela de dimensão DIM_DATA,

e retornar o valor usado para o campo ID_DATA que é chave primaria da tabela e é um campo auto_increment do SGBD MySQL;

7º - DIM_HORA: Step do tipo "*Combinação Lookup / Update"*, responsável por inserir os dados nos campos hora, minuto e segundo na tabela de dimensão DIM_HORA, e retornar o valor usado para o campo ID_HORA que é chave primaria da tabela e é um campo auto_increment do SGBD MySQL;

8º - DIM_RNC: Step do tipo "*Combinação Lookup / Update"*, responsável por inserir os dados no campo nm_rnc na tabela de dimensão DIM_RNC, e retornar o valor usado para o campo ID_RNC que é chave primaria da tabela e é um campo auto_increment do SGBD MySQL;

9º - DIM_NODEB: Step do tipo "*Combinação Lookup / Update"*, responsável por inserir os dados no campo nm_nodeb na tabela de dimensão DIM_RNC, e retornar o valor usado para o campo ID_NODEB que é chave primaria da tabela e é um campo auto_increment do SGBD MySQL;

10º - Cidade e Bairro: Step do tipo "*String Cut"*, responsável extrair a pseudo cidade dos três últimos caracteres do nome da RNC, e o pseudo bairro dos cinco caracteres do meio do nome da NODEB, pois os dados desses campos não estão disponíveis no arquivo CSV;

11º - DIM_GEOGRAFIA: Step do tipo "*Combinação Lookup / Update"*, responsável por inserir os dados nos campos bairro, cidade, e estado na tabela de dimensão DIM_GEOGRAFIA, e retornar o valor usado para o campo ID_GEOGRAFIA que é chave primaria da tabela e é um campo auto_increment do SGBD MySQL. Os campos cidade e bairro foram obtidos no step anterior, mas o campo estado optamos por deixar com o valor nulo;

12º - FATO_CONTADORES: Step do tipo "*Saída a Tabela"*, responsável por inserir os dados nos campos ID_DATA, ID_HORA, ID_RNC, ID_NODEB e ID_GEOGRADIA, que compõem a chave primária da tabela de fatos, além de inserir os valores nos campos dos contadores de 1 a 14;

13º - Chave duplicada: Step do tipo "*Dummy"*, responsável em desprezar as linhas duplicados do step anterior (FATO_CONTADORES).

6.3.1.2 Dificuldades encontradas na montagem

A grande dificuldade encontrada no Kettle foi o fato de cada linha do arquivo exportado ser tratada como uma linha independente em todo o processo, ou seja, a primeira linha exportada já era processada por todos os steps, na devida seqüência, até o final do processo e se não tivéssemos a chave primária da tabela de fatos completa, não poderíamos inserir o registro na tabela de fatos. Isso se caracterizou como uma dificuldade, esse problema tornou necessário uma nova pesquisa sobre o assunto que resultou no step "*Combinação Lookup / Update"*, que simplificou bastante o processo de ETL anteriormente construído. Nesse step é possível obter o valor do campo auto_increment do SGDB MySQL usado como chave primária na inserção de um novo registro, contribuindo assim para um processo mais simples e com uma melhor performance.

6.3.1.3 Facilidades encontradas na montagem

Na ferramenta Kettle a facilidade mais notada foi a possibilidade de a cada step visualizarmos os campos que entram neste step e os campos que saem. Ficando claro a partir de qual step tal campo ficou disponível ou deixou de existir no fluxo de dados.

6.3.1.4 Tempo de execução do processo ETL

Todos os testes foram feitos em um Netbook Asus EeePC com processador Intel Atom(TM) CPU N280 @ 1.66GHz, 1GB Ram, Windows XP Home Edition SP3.

Como não conseguimos montar o processo de ETL com a mesma lógica nas duas ferramentas, e por achar que isso prejudicaria consideravelmente a análise, este item foi dividido em dois cenários, conforme abaixo:

Cenário 1 - Processo de ETL completo, com a carga das tabelas de dimensão e da tabela fato;

Cenário 2 – Processo de ETL parcial, com a carga apenas das tabelas de dimensão.

Na ferramenta Kettle tivemos os seguintes tempos de execução:

Cenário 1:

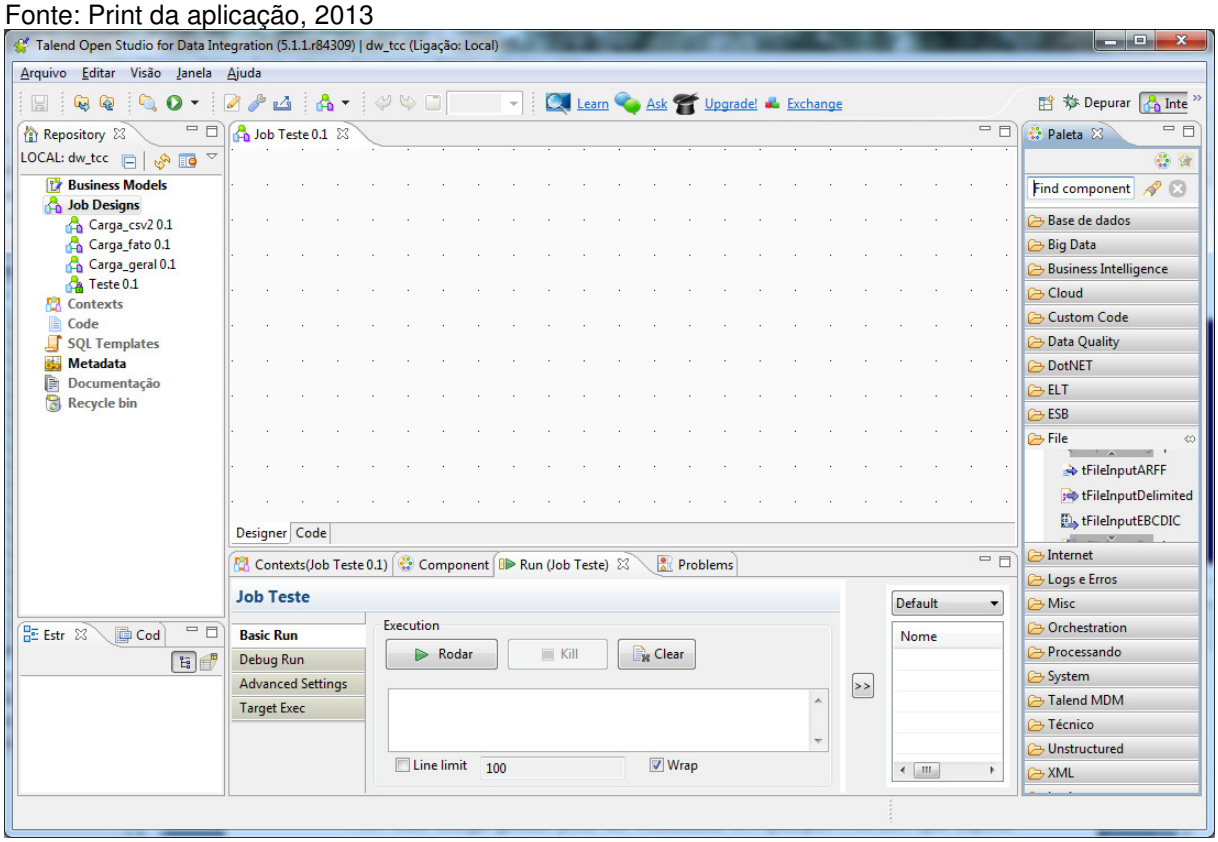
- Em média 14 segundos com a base de dados vazia.
- Em média 40 segundos com a base de dados já povoada.

Cenário 2:

- Em média 5 segundos com a base de dados vazia.
- Em média 3 segundos com a base de dados já povoada.

6.3.2 Talend Open Studio

Figura 12 – Interface do Talend Open Studio, versão 5.1.1
Fonte: Print da aplicação, 2013



A ferramenta *Talend Open Studio* (TOS), que possui a interface gráfica apresentada na Figura 12, é escrita em Java e utiliza a plataforma RCP (*Rich Client Platform*) do Eclipse (Ambiente de Desenvolvimento Integrado reconhecido mundialmente), podendo gerar como resultado final um código Java ou Perl. Este código gerado pode ser visualizado em qualquer interface que suporte uma destas linguagens e executado nos ambientes que suportem as mesmas. Com isso os desenvolvedores podem utilizar seus conhecimentos nas linguagens Java ou Perl para implementarem certas necessidades.

Como padrão do Eclipse o TOS também usa um repositório local, conhecido como "*workspace*", para armazenar todos os projetos e definições dos mesmos. Diante desse cenário precisamos sempre criar um projeto, e dentro dele criar os artefatos necessários para se montar um processo de ETL. O principal artefato é o job, que nos permite criar dentro dele os processos de ETL. Para criar

um processo de ETL temos os vários componentes que estão disponíveis na paleta de componentes, e como exemplo podemos citar o componente “tFileInputDelimited”, que é responsável pela entrada de dados de um arquivo CSV, se assemelhando muito com os steps usados na ferramenta Kettle. Estes componentes também possuem um fluxo de dados, que aqui é conhecido como ”conexão”, e é representado por uma linha com uma seta na ponta do componente de destino. A grande diferença é que essa “conexão” é dividida em muitos tipos e subtipos diferentes. Como exemplo, temos o tipo de conexão “*row”*, que captura os dados de um schema definido no componente que está sendo usado, e conforme o subtipo (main, lookup, filter, etc) usado, trata os dados disponíveis de forma diferente no fluxo.

No TOS o processo de ETL é montado somente nos jobs, diferente do Kettle que possui transformações e jobs. Para que um job chame outro Job num ponto específico do processo, temos o componente “tRunJob”. Portanto podemos montar um job principal que chame, na sequência desejada, os outros jobs necessários para concluir um processo de ETL.

Na interface gráfica temos a opção de executar os jobs criados e visualizar o log de execução, que mostra também os possíveis erros encontrados. A interface ainda disponibiliza as informações da quantidade de linhas processadas, o tempo levado para processar estas linhas e a média de linhas que serão ou foram processadas por segundo, junto à própria conexão entre dois componentes.

Quando abrimos a interface gráfica podemos selecionar o projeto que desejamos abrir, e quando isso é feito a interface precisa montar toda a parte visual do processo, e isso demora bastante, mas é considerado normal, pois a interface a partir de um código gerado em Java ou Perl e monta toda sua parte visual.

6.3.2.1 Montagem do processo de ETL

Figura 13 – Job 1 do processo ETL montado no TOS
Fonte: Elaborado pelo autor, 2013

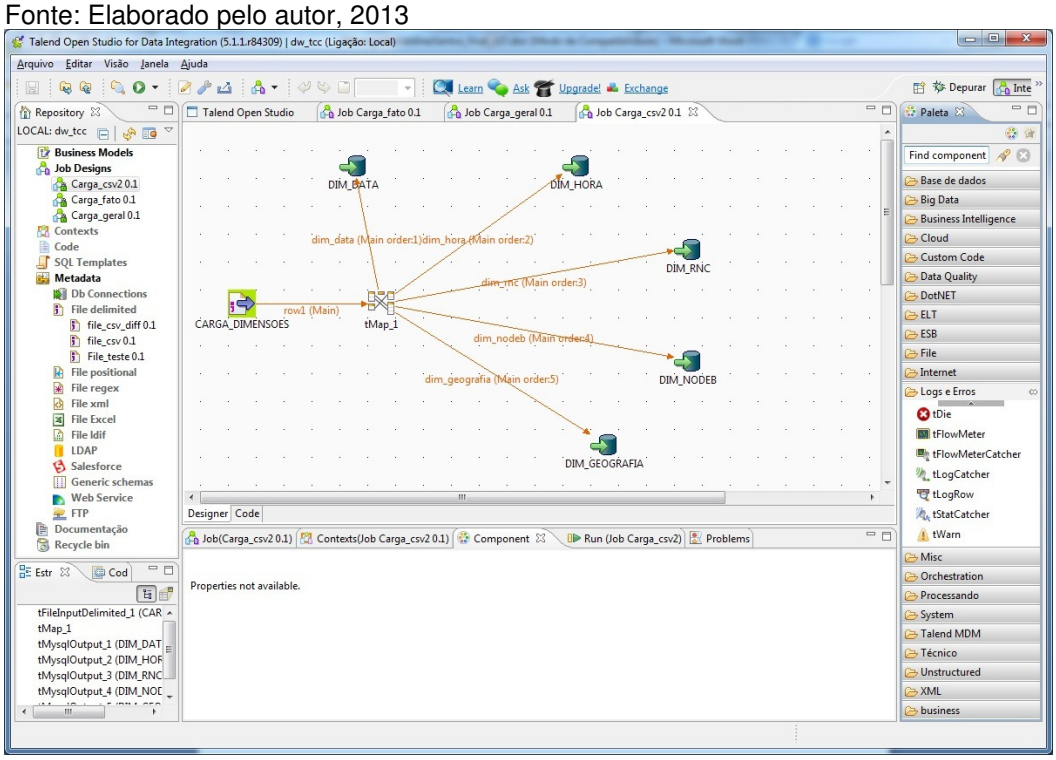


Figura 14 – Job 2 do processo ETL montando no TOS
Fonte: Elaborado pelo autor, 2013

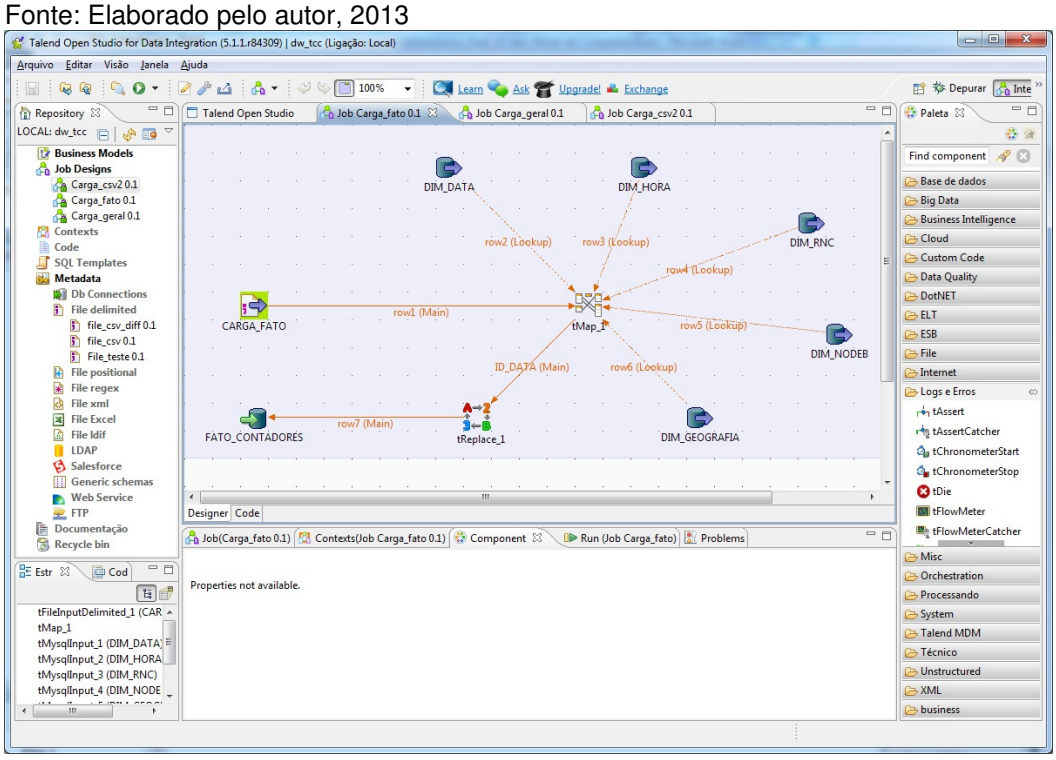
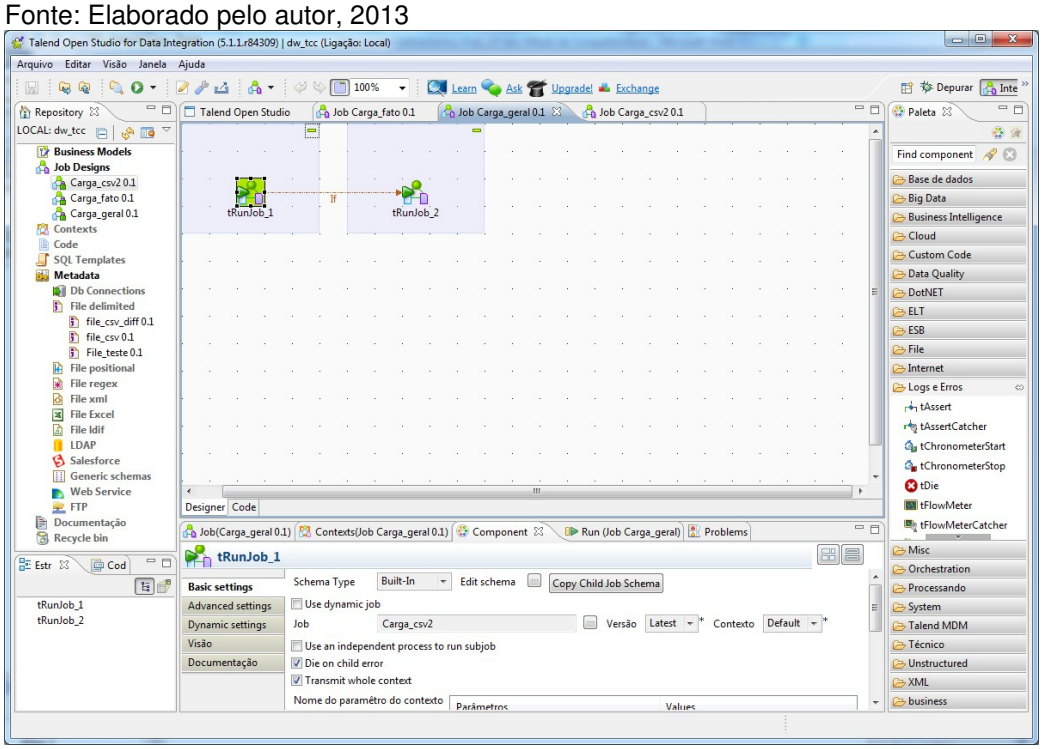


Figura 15 – Job principal do processo ETL montado no TOS
Fonte: Elaborado pelo autor, 2013



Para montar o processo de ETL proposto na ferramenta TOS, foi necessário desenvolver três jobs, sendo o primeiro, apresentado na Figura 13, responsável pela extração dos dados do arquivo CSV e a transformação e inserção destes nas tabelas de dimensão. O segundo job, apresentado na Figura 14, ficou com a função de extrair novamente os dados do arquivo CSV e com base no dado de cada linha buscar nas tabelas de dimensão a chave primária necessária para inserção daquele registro na tabela de fatos. Já o terceiro job, apresentado na Figura 15, ficou responsável apenas pela chamada do primeiro job e após a finalização deste com sucesso, a chamada do segundo job. Caso ocorra algum erro na execução do primeiro job o segundo não será executado.

Os jobs foram montados usando os seguintes componentes disponíveis na ferramenta:

Job 1: Nomeado de “Carga_csv2”.

1º - CARGA DIMENSOES: Componente do tipo “tFileInputDelimited”, responsável pela extração dos dados do arquivo CSV;

2º - tMap: Componente responsável pelo mapeamento de todas as colunas extraídas do arquivo CSV para as respectivas tabelas de dimensão, fazendo as transformações necessárias antes da inserção;

3º - DIM_DATA: Componente do tipo “tMySQLOutput”, responsável por inserir os dados dos campos dia, mês e ano na tabela de dimensão DIM_DATA. Os campos citados são gerados no componente “tMap”, com a extração das respectivas substrings da coluna que contem a data do registro;

4º - DIM_HORA: Componente do tipo “tMySQLOutput”, responsável por inserir os dados dos campos hora, minuto e segundo na tabela de dimensão DIM_HORA. Os campos citados são gerados no componente “tMap”, com a extração das respectivas substrings da coluna que contem à hora do registro, sendo o campo segundo fixado em 00;

5º - DIM_RNC: Componente do tipo “tMySQLOutput”, responsável por inserir os dados do campo nm_rnc na tabela de dimensão DIM_RNC. O campo citado é gerado no componente “tMap”, com um simples mapeamento;

6º - DIM_NODEB: Componente do tipo “tMySQLOutput”, responsável por inserir os dados do campo nm_nodeb na tabela de dimensão DIM_NODEB. O campo citado é gerado no componente “tMap”, com um simples mapeamento;

7º - DIM_GEOGRAFIA: Componente do tipo “tMySQLOutput”, responsável por inserir os dados dos campos bairro, cidade e estado na tabela de dimensão DIM_GEOGRAFIA. Os campos cidade e bairro foram gerados no componente “tMap”, com a extração de certas substrings do campo nm_rnc e nm_nodeb, e deixando o valor do campo estado como nulo;

Job 2: Nomeado de “Carga_fato”.

1º - CARGA_FATO: Componente do tipo “tFileInputDelimited”, responsável pela extração dos dados do arquivo CSV;

2º - tMap: Componente responsável pelo mapeamento de todas as chaves primárias das tabelas de dimensão, que entram neste componente, e correspondem aos dados vindos do arquivo CSV lido no componente “CARGA_FATO”;

3º - DIM_DATA: Componente do tipo “tMySQLInput”, responsável pela extração dos dados da tabela de dimensão DIM_DATA. Todos os campos são enviados para o componente “tMap”;

4º - DIM_HORA: Componente do tipo “tMySQLInput”, responsável pela extração dos dados da tabela de dimensão DIM_HORA. Todos os campos são enviados para o componente “tMap”;

5º - DIM_RNC: Componente do tipo “tMySQLInput”, responsável pela extração dos dados da tabela de dimensão DIM_RNC. Todos os campos são enviados para o componente “tMap”;

6º - DIM_NODEB: Componente do tipo “tMySQLInput”, responsável pela extração dos dados da tabela de dimensão DIM_NODEB. Todos os campos são enviados para o componente “tMap”;

7º - DIM_GEOGRAFIA: Componente do tipo “tMySQLInput”, responsável pela extração dos dados da tabela de dimensão DIM_GEOGRAFIA. Todos os campos são enviados para o componente “tMap”;

8º - tReplace: Componente responsável pela substituição dos valores que vem do arquivo CSV como nulo (null), pelo valor zero, para que não ocorra erro na conversão do tipo String para Double. Este componente foi necessário para corrigir um erro que ocorria no processo;

9º - FATO_CONTADORES: Componente do tipo “tMySQLOutput”, responsável pela carga dos contadores de 1 a 14 na tabela de fato, junto à chave primária composta pelas chaves primárias de todas as tabelas de dimensão.

Job 3: Nomeado de “Carga_geral”.

1º - tRunJob_1: Componente responsável por chamar o Job “Carga_csv2” para execução;

2º - tRunJob_2: Componente responsável por chamar o Job “Carga_fato” para execução. Devido o tipo de conexão “Run if” com o componente “tRunJob_1”, este só será executado se o componente anterior for executado com sucesso.

6.3.2.2 Dificuldades encontradas na montagem

No TOS encontramos dificuldade para acompanhar o fluxo de dados. No Kettle temos a possibilidade de visualizar a cada step o que entra e o que sai daquele step específico, e no TOS só conseguimos acompanhar o fluxo se jogarmos as informações daquele componente para um componente de log, o que dificultou bastante a detecção dos erros que enfrentamos no processo.

Outra dificuldade foi o fato de não conseguirmos encontrar nenhum componente que inserisse os dados das dimensões no SGDB MySQL, e retornasse o valor do campo chave primária, que é um campo auto_increment, para ser usado no fluxo. Com isso tivemos que montar dois jobs, um para inserir as dimensões e outro para inserir na tabela de fatos, fazendo assim com que o arquivo CSV fosse lido duas vezes e que os dados das tabelas de dimensão também fossem lidos.

6.3.2.3 Facilidades encontradas na montagem

Uma facilidade encontrada foi na inserção dos dados nas tabelas de dimensão. Como o TOS possui o componente “tMap”, que possibilita entrarmos com uma fonte de dados (arquivo CSV) e dele mapear para varias tabelas de destino (tabelas de dimensão), fazendo o uso de diversas funções do TOS para manipular estes dados. Com isso o cenário de inserção dos dados nas tabelas de dimensão ficou muito simples.

Outra facilidade encontrada foi o artefato “*metadata*”, que possibilita a criação de padrões, como de arquivos CSV, arquivos Excel, Web service, conexões de banco de dados, entre outros. Estes padrões podem ser usados em componentes específicos e facilitam muito a manutenção, que neste caso fica centralizada em um único ponto. Diferente de quando fizemos estas configurações nos próprios componentes, o que requer a alteração em vários pontos, caso algum ajuste seja necessário.

6.3.2.4 Tempo de execução do processo ETL

Considerando o já exposto no primeiro e segundo parágrafos do item 6.3.1.4, segue os tempos de execução levantados na ferramenta TOS:

Cenário 1:

- Em média 40 segundos com a base de dados vazia.

- Em média 40 segundos com a base de dados já povoada.

Cenário 2:
- Em média 27 segundos com a base de dados vazia.
- Em média 25 segundos com a base de dados já povoada.

6.3.3 Interpretação dos dados

Como resultado da análise feita, construímos dois quadros. O Quadro 1, mostra um comparativo entre alguns dados levantados sobre as duas ferramentas analisadas. Já o Quadro 2, nos mostra um comparativo com as principais informações levantadas nos testes efetuados.

Quadro 1 – Informações sobre as ferramentas

Ferramenta	Kettle	Talend Open Studio
Sistema Operacional	Windows, Linux e Unix	Windows, Linux e Unix
Versão	4.1.0	5.1.0
Linguagem de desenvolvimento	Java.	Java.
Ambiente gráfico	Baseado no SWT.	Baseado no Eclipse.
Característica principal do ambiente gráfico	Usa Jobs para construção dos processos ETL.	Usa Jobs e Transformações para construção dos processos ETL.
Produto final gerado pela interface gráfica	Arquivo em formato XML para ser interpretado pelas aplicações Kitchen ou Pan.	Programa em Java ou Perl, para serem executados em qualquer ambiente que suporte estas linguagens.

Fonte: Elaborado pelo Autor, 2013

No Quadro 2 temos dois itens categorizados por níveis (**fácil**, **médio**, **difícil**). Níveis estes que levam em consideração um usuário com conhecimento básico em banco de dados e análise de sistemas. Com isso definimos o seguinte:

- Fácil: Não terá grandes problemas;
- Médio: Enfrentará alguns problemas;
- Difícil: Terá grandes problemas.

Estes itens foram criados devido à dificuldade encontrada em contabilizar o tempo gasto para construção do processo de ETL em ambas as ferramentas, pois esse tempo sempre estará diretamente relacionado com o nível de conhecimento do usuário em cada ferramenta. Considerando neste caso que o autor desta monografia já possuía conhecimentos medianos na ferramenta Kettle e nenhum conhecimento na ferramenta TOS.

Quadro 2 – Comparativo entre as ferramentas

	Kettle	**Talend Open Studio**
Tempo execução cenário 1. (Base vazia)	14	40
Tempo execução cenário 1. (Base povoada)	40	40
Tempo execução cenário 2. (Base vazia)	5	27
Tempo execução cenário 2. (Base povoada)	3	25
Nível de facilidade na construção do processo de ETL.	Médio	Difícil
Nível de facilidade no diagnóstico de problemas na montagem do processo de ETL.	Fácil	Médio

Fonte: Elaborado pelo autor, 2013

Conforme apresentado no Quadro 2, o tempo que cada ferramenta levou para executar o processo de ETL, levando em conta os dois cenários propostos, e considerando que o segundo cenário teve como foco deixar o processo nas duas ferramentas o mais idêntico possível, nos mostrou que o Kettle é mais eficiente na execução de processos que tenham como foco principal a extração de dados de um arquivo CSV e inserção destes em tabelas de um SGBD MySQL, tanto numa carga full como numa carga de dados incremental. Havendo um ligeiro empate na execução do primeiro cenário, onde a base de dados já estava povoada com os mesmos dados, forçando assim que a carga desprezasse todos os registros.

Já em relação ao tempo de desenvolvimento do processo de ETL, o Kettle se mostrou mais eficiente, pois deixa o usuário com um maior controle sobre o fluxo de dados, sendo possível visualizar o que está ocorrendo após cada step. O fato de cada step ser responsável por uma transformação específica também colabora bastante para o entendimento do que está sendo feito naquele ponto do processo. Considerando também que a interface Spoon, do Kettle, é mais simples de usar do que a interface do TOS, o que leva o usuário iniciante a ter uma curva de aprendizagem menor para se criar um processo simples de ETL. Lembrando é claro, que o autor desta monografia já possuía conhecimentos medianos na ferramenta Kettle, o que pode ter contribuído para uma leve distorção nesta conclusão.

Para diagnosticar problemas o Kettle também se saiu melhor, pois a facilidade em acompanhar o fluxo de dados, a cada step, facilita o diagnóstico da origem de um possível erro que esteja ocorrendo no final ou no meio do processo, por exemplo.

Levando em conta os cenários montados na análise, tivemos a percepção que a ferramenta Kettle terá uma melhor produtividade. Mas em outros cenários a ferramenta TOS também poderá se mostrar mais eficiente, pois tem muito potencial para isso. Temos a convicção que tanto uma ferramenta como a outro poderão ser a melhor escolha para determinado projeto, e só uma comparação do cenário específico em cada ferramenta que nos comprovará qual a melhor.

7 CONCLUSÃO

Com o desenvolvimento deste trabalho concluímos que analisar a performance de ferramentas de ETL é algo muito complexo. Cada ferramenta possui suas características próprias e permitem que um determinado processo de ETL seja feito de várias formas diferentes, usando vários steps ou componentes distintos, conforme a ferramenta usada, e principalmente conforme o conhecimento de cada usuário, que influenciará na construção de um processo mais otimizado ou não.

Em relação aos testes, ficou evidente que as duas ferramentas possuem uma gama muito grande de recursos e atendem tranquilamente qualquer necessidade dos processos de ETL. Além de ambas possuírem ainda a opção da criação de steps ou componentes conforme a necessidade específica, usando os recursos disponíveis na linguagem de programação Java, e também estarem sempre em evolução pela comunidade *Open Source*, que lança constantemente novas versões com novos recursos.

Mesmo com todos esses testes e conclusões pessoais, e reafirmando que o conhecimento sobre cada ferramenta é fundamental para que se possa obter uma melhor performance, temos como ponto forte a constatação que cada ferramenta analisada possui suas características específicas, e não existe uma ferramenta melhor do que outra, pois a melhor ferramenta é aquela na qual o usuário se sente mais confortável e retém um maior conhecimento sobre os seus recursos, para assim montar os processos de ETL da melhor maneira possível.

REFERÊNCIAS BIBLIOGRÁFICAS

BARBIERI, Carlos. **BI – Business Intelligence:** Modelagem & Tecnologia. Rio de Janeiro: Axcel, 2001.

CEDET. **Node-B**. Disponível em: <http://www.cedet.com.br/index.php?/O-que-e/3G-Wireless/node-b.html>. Acesso em: 13 Dez. 2012.

TECHNET. **SQL Server 2000 Technical Articles**. Disponível em:<http://technet.microsoft.com/en-us/library/aa902672(v=sql.80).aspx>. Acesso em: 20 Set. 2012.

GARTNER GROUP. **Magic Quadrant**. Disponível em: <http://www.gartner.com>. Acesso em: 24 Ago. 2012.

INMON, W. H. **Como construir o Data Warehouse**. 2. ed. Rio de Janeiro: Campos, 1997.

KIMBALL, R. **The Data Warehouse Toolkit:** guia completo para modelagem dimensional. Rio de Janeiro: Campus, 2002.

PENTAHO KETTLE PROJECT. **Pentaho Data Integration (Kettle)**. Disponível em: <http://kettle.pentaho.com>. Acesso em: 28 Set. 2012.

SEZÕES, Carlos; OLIVEIRA, José; BAPTISTA Miguel. **Business Intelligence**. Porto: Sociedade Portuguesa de Inovação, 2006.

PRIVATI, Carlos Eduardo Andriotti. Talend Open Studio: Uma ferramenta open source de integração de dados e ETL – Parte 1. **SQL Magazine**: Grajaú, 71, 55-63, 2009.

PRIVATI, Carlos Eduardo Andriotti. Talend Open Studio: Uma ferramenta open source de integração de dados e ETL – Parte 2. **SQL Magazine**: Grajaú, 72, 55-62, 2010.

TALEND. **Data Integration**. Disponível em : <http://www.talend.com/products/data-integration>. Acesso em: 29 Set. 2012.

THE BI VERDICT. **Storing multidimensional data:** Relation table layouts. Disponível em: <http://www.bi-verdict.com/fileadmin/dl_temp/96003cfa8a75e5d3c48f2641edf37109/multidimensional_storage.htm>. Acessado em: 20 Set. 2012

WIKIPEDIA. **UMTS Terrestrial Radio Access Network**. Disponível em: <http://en.wikipedia.org/wiki/UMTS_Terrestrial_Radio_Access_Network>. Acesso em: 13 Dez. 2012.

APÊNDICES

APÊNDICE A – Script de criação das bases de dados usadas para testes

Como a base criada para as duas ferramentas era idêntica, somente as duas primeiras linhas que mudam de um script para o outro.

Para Kettle:

```
create database dw_kettle;
use dw_kettle;
```

Para Talend Open Studio:

```
create database dw_talend;
use dw_talend;
```

Comum para as duas bases de dados:

```
CREATE TABLE DIM_NODEB (
  id_nodeb INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  nm_nodeb VARCHAR(20) NULL,
  PRIMARY KEY(id_nodeb),
  UNIQUE INDEX uk_DIM_NODEB(nm_nodeb)
);


CREATE TABLE DIM_RNC (
  id_rnc INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  nm_rnc VARCHAR(20) NULL,
  PRIMARY KEY(id_rnc),
  UNIQUE INDEX uk_DIM_RNC(nm_rnc)
);


CREATE TABLE DIM_HORA (
  id_hora INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  hora SMALLINT UNSIGNED NULL,
  minuto SMALLINT UNSIGNED NULL,
  segundo SMALLINT UNSIGNED NULL,
  PRIMARY KEY(id_hora),
  UNIQUE INDEX uk_DIM_HORA(hora, minuto, segundo)
);



CREATE TABLE DIM_DATA (
  id_data INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  dia SMALLINT UNSIGNED NULL,
  mes SMALLINT UNSIGNED NULL,
  ano SMALLINT UNSIGNED NULL,
  PRIMARY KEY(id_data),
  UNIQUE INDEX uk_DIM_DATA(dia, mes, ano)
);
```

```
CREATE TABLE DIM_GEOGRAFIA (
  id_geografia INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  bairro VARCHAR(45) NULL,
  cidade VARCHAR(45) NULL,
  estado VARCHAR(45) NULL,
  PRIMARY KEY(id_geografia),
  UNIQUE INDEX uk_DIM_GEOGRAFIA(bairro, cidade)
);


CREATE TABLE FATO_CONTADORES (
  ID_DATA INTEGER UNSIGNED NOT NULL,
  ID_HORA INTEGER UNSIGNED NOT NULL,
  ID_RNC INTEGER UNSIGNED NOT NULL,
  ID_NODEB INTEGER UNSIGNED NOT NULL,
  ID_GEOGRAFIA INTEGER UNSIGNED NOT NULL,
  contador1 DECIMAL(10,3) NULL,
  contador2 DECIMAL(10,3) NULL,
  contador3 DECIMAL(10,3) NULL,
  contador4 DECIMAL(10,3) NULL,
  contador5 DECIMAL(10,3) NULL,
  contador6 DECIMAL(10,3) NULL,
  contador7 DECIMAL(10,3) NULL,
  contador8 DECIMAL(10,3) NULL,
  contador9 DECIMAL(10,3) NULL,
  contador10 DECIMAL(10,3) NULL,
  contador11 DECIMAL(10,3) NULL,
  contador12 DECIMAL(10,3) NULL,
  contador13 DECIMAL(10,3) NULL,
  contador14 DECIMAL(10,3) NULL,
  PRIMARY KEY(ID_DATA, ID_HORA, ID_RNC, ID_NODEB, ID_GEOGRAFIA),
  INDEX FATO_CONTADORES_FKIndex1(ID_DATA),
  INDEX FATO_CONTADORES_FKIndex2(ID_HORA),
  INDEX FATO_CONTADORES_FKIndex3(ID_RNC),
  INDEX FATO_CONTADORES_FKIndex4(ID_GEOGRAFIA),
  INDEX FATO_CONTADORES_FKIndex5(ID_NODEB),
  FOREIGN KEY(ID_DATA)
    REFERENCES DIM_DATA(id_data)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  FOREIGN KEY(ID_HORA)
    REFERENCES DIM_HORA(id_hora)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  FOREIGN KEY(ID_RNC)
    REFERENCES DIM_RNC(id_rnc)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  FOREIGN KEY(ID_GEOGRAFIA)
    REFERENCES DIM_GEOGRAFIA(id_geografia)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  FOREIGN KEY(ID_NODEB)
    REFERENCES DIM_NODEB(id_nodeb)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
);
```

APÊNDICE B – Amostra de 9 linhas do arquivo CSV fornecido como fonte de dados

```
;RNC01TRL    ;3GRNPTB63315
;06/07/2012;01:00;      190356.000;        3600.000;        -106.812;
37.904;          -99.800;        -107.100;          42.494;           8.046;
4.255;           38.911;           36.010;          36.670;           7.922;
4.255;
;RNC01TRL    ;3GRNPTB63315
;06/07/2012;02:00;      185766.000;        3600.000;        -106.940;
37.693;         -104.800;        -107.200;          42.638;           8.025;
4.255;           38.682;           35.788;          36.578;           7.913;
4.255;
;RNC01TRL    ;3GRNPTB63315
;06/07/2012;03:00;      185508.000;        3600.000;        -106.947;
37.526;         -104.600;        -107.200;          42.685;           8.008;
4.255;           38.315;           35.788;          36.462;           7.901;
4.255;
;RNC01TRL    ;3GRNPTB63314
;06/07/2012;01:00;      190241.000;        3600.000;        -106.816;
36.694;         -106.000;        -107.100;          42.731;           7.925;
4.255;           37.624;           35.788;          36.155;           7.871;
4.255;
;RNC01TRL    ;3GRNPTB63314
;06/07/2012;02:00;      188920.000;        3600.000;        -106.852;
38.325;         -105.600;        -107.000;          42.542;           8.088;
4.255;           37.150;           35.788;          36.065;           7.862;
4.255;
;RNC01TRL    ;3GRNPTB63314
;06/07/2012;03:00;      189297.000;        3600.000;        -106.842;
38.776;         -103.700;        -107.000;          42.685;           8.133;
4.255;           37.472;           35.788;          36.031;           7.858;
4.255;
;RNC01TRL    ;3GRNARP63796
;06/07/2012;01:00;      177524.000;        3598.000;        -107.166;
40.457;         -100.600;        -107.400;          42.731;           8.301;
4.255;           38.052;           36.010;          36.323;           7.888;
4.255;
;RNC01TRL    ;3GRNARP63796
;06/07/2012;02:00;      173003.000;        3598.000;        -107.292;
37.517;         -104.800;        -107.500;          41.976;           8.007;
4.255;           37.624;           35.788;          36.151;           7.870;
4.255;
;RNC01TRL    ;3GRNARP63796
;06/07/2012;03:00;      171596.000;        3598.000;        -107.331;
40.209;         -103.800;        -107.500;          42.685;           8.276;
4.255;           36.802;           35.788;          36.084;           7.864;
4.255;
```