

Edmir de Jesus Oliveira Tavares

O Processo ETL

O Caso da Unitel T+ Telecomunicações

Universidade Jean Piaget de Cabo Verde

Campus Universitário da Cidade da Praia
Caixa Postal 775, Palmarejo Grande
Cidade da Praia, Santiago
Cabo Verde

3.11.13

Edmir de Jesus Oliveira Tavares

Processo ETL

O Caso da Unitel T+ Telecomunicações

Universidade Jean Piaget de Cabo Verde

Campus Universitário da Cidade da Praia
Caixa Postal 775, Palmarejo Grande
Cidade da Praia, Santiago
Cabo Verde

3.11.13

Edmir de Jesus Oliveira Tavares, autor da monografia intitulada Processo ETL – caso da Unitel T+ Telecomunicações, declaro que, salvo fontes devidamente citadas e referidas, o presente documento é fruto do meu trabalho pessoal, individual e original.

Cidade da Praia, 22 de Outubro de 2013

Edmir de Jesus Oliveira Tavares

Memória Monográfica apresentada à Universidade Jean Piaget de Cabo Verde como parte dos requisitos para a obtenção do grau de Licenciatura em Engenharia de Sistemas e Informática.

Sumário

O trabalho que se segue fala sobre o processo ETL (*Extract, Transform and Load*) e as ferramentas que estão associadas ao ETL. É apresentado um enquadramento teórico sobre esse processo, onde são distinguidas as principais etapas desse processo (Extração, Transformação e *Load*) e aprofundar um pouco sobre esse conceito. Fala também sobre as ferramentas de ETL (Comerciais e *OpenSource*), com destaque para a ferramenta *Talend Open Studio for Data Integration* visto que ela é utilizada para implementação de sistemas de ETL na Unitel T+ e vai ser apresentado um estudo do caso prático sobre esses sistemas.

Palavras-chave: ETL, Processo ETL, Ferramenta ETL, *OpenSource*, *Extract Transform and Load*, *Talend Open Studio for Data Integration*, Extração, Transformação e *Load*

Dedicatórias

Á minha mãe Dulce António de Oliveira por todo apoio que me deu ao longo da vida para que eu seja sempre uma boa pessoa, e consiga vencer mais essa fase da vida.

Ao meu filho Omário Santos de Pina Oliveira por me conceder mais um motivo de continuar a luta.

Aos meus irmãos que fazem parte de min.

Dedico este trabalho á minha família.

Agradecimentos

Primeiramente agradeço a Deus, que me concedeu a vida e saúde e esta oportunidade.

Agradeço a minha mãe Dulce António de Oliveira que sempre me ajudou com o pouco que podia, sempre com dedicação e amor, agradeço também aos meus tios e tias. Agradeço também á Josiana Tavares de Pina Monteiro pelo apoio concedido durante todo esse tempo.

Agradeço o meu Professor, Orientador Doutor Isaías Barreto da Rosa, que me tem ajudado muito na realização desse trabalho, nos estudos e por me dar uma oportunidade. Agradeço também aos meus professores pelos conhecimentos que partilharam e dedicação.

Aos meus colegas de trabalho pelo apoio que me deram para conclusão desse trabalho e pelo incentivo.

Á todos que de alguma forma contribuíram para o meu sucesso e crescimento, principalmente na conclusão desse trabalho.

Um grande Obrigado a todos.

Conteúdo

Introdução.....	19
1 Objectivos	21
2 Motivação	22
3 Metodologia.....	22
4 Estrutura do documento.....	23
5 Limitações do estudo	23
Capítulo 1: Processo ETL	25
1 Introdução	25
2 Conceito de ETL.....	25
3 História de ETL	28
4 Requisitos de um ETL	31
5 As etapas do Processo ETL	33
5.1 A Extração de dados	34
5.1.1 Método de extração Lógica	35
5.1.2 Método de extração Física.....	36
5.2 Transformação e Limpeza de dados	37
5.2.1 Funções da transformação de dados	39
5.3 <i>Load</i> ou Inserção de dados	42
6 Integração de Dados	45
6.1 Enquadramento Histórico	46
6.2 Abordagem para Integração de Dados.....	47
6.2.1 Consolidação de Dados	48
6.2.2 Federação de Dados.....	49
6.2.3 Propagação de Dados.....	50
6.2.4 Comparação das três abordagens.....	52
7 Os Subsistemas de ETL.....	53
7.1 Extração: Coletar os Dados	54
7.2 Limpar e Ajustar os Dados	55
7.3 Entrega: Preparação para Apresentação	56
7.4 Gerenciamento do Ambiente de ETL.....	58
8 Os Metadados	60
8.1 O que são Metadados.....	61

8.2	Metadados ETL	62
Capítulo 2: As Ferramentas de ETL.....		64
1	Introdução	64
2	Conceito.....	64
3	A Evolução das Ferramentas de ETL	66
3.1	Primeira Geração de ETL: Geradores de Códigos	67
3.2	Segunda Geração de ETL: Mecanismos de ETL.....	70
3.3	Terceira Geração de ETL: A Arquitetura de ETL	73
4	As Ferramentas de ETL Disponíveis no Mercado.....	77
4.1	As mais conhecidas	80
4.2	<i>As Open Source</i>	83
4.2.1	O que significa Open Source	83
4.2.2	As Ferramentas de ETL Open Source	84
5	Comprar ou Desenvolver?	85
5.1	Codificação Manual.....	90
5.2	Ferramentas de ETL	91
6	A Ferramenta Talend Open Studio para Integração de Dados	93
6.1	História	95
6.2	Benefícios	96
Capítulo 3: Caso Prático – Uso da Ferramenta TOSDI para integração e controlo dos dados		97
1	Enquadramento	97
2	Apresentação da empresa	97
2.1	Organização Geral	98
3	Integração de Dados na Unitel T+	99
3.1	Histórico dos métodos ou ferramentas utilizadas para ID	99
3.1.1	Codificação Manual.....	100
3.1.2	Uso de Ferramentas ETL.....	103
3.2	O Problema	104
3.3	A solução	106
4	O sistema desenvolvido	107
4.1	As Fontes de Dados	108
4.2	Tabelas criadas	109
4.3	O projeto desenvolvido no Talend	112
4.3.1	Job dataFileError	113

4.3.2	Job dataConfigFile.....	114
4.3.3	Job dataListFTPLastNDir.....	115
4.3.4	Job dataListFileDir	115
4.3.5	Job dataExtractFile	116
4.3.6	Job dataLoadWithControl	117
4.4	A avaliação do Sistema.....	120
4.5	Comparação entre o Antes e o Depois.....	121
5	Considerações finais	122
Conclusão.....		123
A	Guião de entrevista.....	132
B	Os entrevistados.....	133

Lista de Tabelas

Tabela 1 - As várias gerações de ETL ao longo dos anos.	29
Tabela 2 - Comparação Incremental e Full Load.	44
Tabela 3 - Comparação das três abordagens para <i>ID</i>	53
Tabela 4 - Lista das Ferramentas de Integração de Dados	80
Tabela 5 - Lista das Ferramentas ETL mais conhecidos	81
Tabela 6 - Ferramenta OpenSource para ID	84
Tabela 7 – Comparação entre o sistema antigo e o atual.....	121

Lista de figuras

Figura 1 - Ambiente do processo ETL	26
Figura 2 - O ciclo de ETL.....	33
Figura 3 - Representação básica para transformação de campo único	41
Figura 4 - Transformação de diversos campos.....	41
Figura 5 - A última etapa do Processo ETL	42
Figura 6 - Integração de Dados	46
Figura 7 - Arquitetura de Consolidação dos Dados e as Tecnologias que a apoiam.....	48
Figura 8 - Arquitetura de Virtualização dos Dados e as Tecnologias que a apoiam.....	49
Figura 9 - Arquitetura de Propagação de Dados e as Tecnologias que a apoiam.....	51
Figura 10 - Ferramentas baseadas em códigos	68
Figura 11 - Ferramentas baseadas em Mecanismos	71
Figura 12 - Quadrante Mágico para Ferramenta de ID 2012	82
Figura 13 - Talend Open Studio for Data Integration	94
Figura 14 - História da companhia Talend	95
Figura 15 - Estrutura Organizacional da Unitel T+	98
Figura 16 - Processamento dos ficheiros no início.....	101
Figura 17 - Controlo inicial dos ficheiros:	102
Figura 18 – Árvore FTP	104
Figura 19 – Relacionamento entre tabela de controlo e de dados	106
Figura 20 – Controlo dos ficheiros	107
Figura 21 - TB_Control_DATA	110
Figura 22 - Schema da tabela TB_DATA	111
Figura 23 - Atributos da tabela TB_FileError	112
Figura 24 - Fluxo do Job dataFileError	113
Figura 25 - Fluxo do Job dataConfigFile	114

Figura 26 - Fluxo do Job dataListFTPLastNDir.....	115
Figura 27 - Fluxo do Job dataListFileDir	116
Figura 28 - Fluxo do Job dataExtractFile	117
Figura 29 - Fluxo do Job dataLoadWithControl	119

Glossário

Arquivo - Organização responsável por gerir, descrever, armazenar e garantir o acesso à informação.

Base de dados - É um sistema cujo objectivo é registar, actualizar, manter e disponibilizar informação relevante para a actividade de uma organização.

Debug – em português “Depuração”, que significa testar um programa, localizar e corrigir quaisquer falhas ou erros. É o processo de encontrar e reduzir defeitos num aplicativo de *software* ou mesmo em hardware. Erros de *software* incluem aqueles que previnem o programa de ser executado e aqueles que produzem um resultado inesperado.

Flat File - São ficheiros simples como ficheiros texto (.txt), ficheiros CSV (*Comma-Separated Values*).

Pipeline – é um conjunto de elementos ligados em série, onde a saída de um elemento é a entrada do próximo num processamento de dados. É como um canal ou um tubo.

Fluxo de Trabalho – do inglês “*Workflow*”, significa a modelização e a gestão informática do conjunto das tarefas a realizar e dos diferentes atores implicado na realização de um processo do negócio.

Processo do Negócio - um conjunto de etapas que uma área de negócio desempenha para criar valor aos seus clientes e também à própria organização. É composto de três elementos básicos, as **Entradas**, as **Atividades** e as **Saídas**.

TimeOuts – significa tempo esgotado, ou sinal que um dispositivo emite quando chegou o tempo limite, no qual estava esperando receber uma informação.

Snapshots - permitem criar imagens de um sistema específico num determinado instante de tempo.

Log - descreve o processo de registro de eventos relevantes num sistema computacional.

Ficheiros *Dump* – ficheiros de *backup* ou uma cópia instantânea de algumas ou todas as informações.

Staging Area – é uma área de armazenamento intermediário entre as fontes de informação e o Data Warehouse (DW) ou Data Mart (DM). É geralmente de natureza temporária, e seu conteúdo pode ser apagado após o DW/DM ter sido carregado com sucesso.

Schema – é uma coleção de objetos de uma base de dados que estão disponíveis para um determinado utilizador ou grupo. Os objetos de um esquema são estruturas lógicas que se referem diretamente aos dados da base de dados. Eles incluem estruturas, tais como tabelas, visões, sequências, procedimentos armazenados, sinónimos, índices, agrupamentos e *links* de base de dados.

Truncate – o comando TRUNCATE remove rapidamente todas as linhas de um conjunto de tabelas.

Flags – serve como uma sinalização, para saber qual um determinado estado.

Lógica *Fuzzy* – é uma extensão da lógica booleana que admite valores lógicos intermediários entre o FALSO (0) e o VERDADEIRO (1). É a lógica que suporta os modos de raciocínio que são aproximados ao invés de exatos.

Breakpoint – o ponto dentro de um programa onde a execução será interrompida para que o programador possa examinar o *status* do programa, o conteúdo das variáveis, e assim por diante.

Batch – é um termo referente a um processamento de dados que ocorre através de um lote de tarefas enfileiradas, de modo que o sistema operacional só processa a próxima tarefa após o término completo da tarefa anterior.

Thread – é uma forma de um processo dividir a si mesmo em duas ou mais tarefas que podem ser executadas concorrentemente.

COBOL – é uma das linguagens de programação mais antigas. O seu nome é a sigla de *COmmon Business Oriented Language* (Linguagem Orientada aos Negócios), que

define seu objetivo principal em sistemas comerciais, financeiros e administrativos para empresas e governos.

Mainframe – é um computador de grande porte, dedicado normalmente ao processamento de um volume grande de informações.

Hub – é o processo pelo qual se transmite ou difunde determinada informação, tendo, como principal característica, que a mesma informação está sendo enviada para muitos receptores ao mesmo tempo.

Roaming - a possibilidade para os clientes sem fio para automaticamente fazer e receber chamadas de voz, enviar e receber dados, ou acessar outros serviços, quando viajam para fora da área de cobertura geográfica da sua própria rede, por meio do uso de uma rede visitada.

Backup - é a cópia de dados de um dispositivo de armazenamento a outro para que possam ser restaurados em caso da perda dos dados originais, o que pode envolver apagamentos acidentais ou corrupção de dados.

Job – é um *design* gráfico de um ou mais componentes ligados entre si, que permite configurar e executar processos de gestão de fluxos de dados. Traduz as necessidades do negócio em códigos, rotinas e programas.

Commit – em SQL é um comando que finaliza uma transação dentro de um sistema de gerenciamento de base de dados (SGBD) e torna visível aos utilizadores todas as alterações.

Siglas e Acrónimos

BD	Base de Dados
BPM	<i>Business Process Management</i>
BI	<i>Business Intelligence</i>
CBS	<i>Convergent Billing Solution</i>
CDC	<i>Change Data Capture</i>
CDR	<i>Call Detail Record</i>
CRC	<i>Cyclic Redundancy Checksum</i>
CSV	<i>Comma-Separated Values</i>
DAS	<i>Data Staging Area</i>
DBA	<i>Database Administrator</i>
DBMS	<i>Data Base Management System</i>
DM	Data Mart
DTS	<i>Data Transformation Services</i>
DW	Data Warehouse
EAI	<i>Enterprise Application Integration</i>
ECM	<i>Enterprise Content Management</i>
EDR	<i>Enterprise Data Replication</i>
EDW	<i>Enterprise Data Warehouse</i>
EII	<i>Enterprise Information Integration</i>
ERP	<i>Enterprise Resource Planning</i>

ESB	<i>Enterprise Service Bus</i>
ETL	<i>Extract Transform Load</i>
FTP	<i>File Transfer Protocol</i>
IBM	<i>International Business Machines</i>
ID	Integração de Dados
IT	<i>Information Technology</i>
MDM	<i>Master Data Management</i>
MSC	<i>Mobile Switching Center</i>
OLAP	<i>On-line Analytical Processing</i>
OSI	<i>Open Source Initiative</i>
OWB	<i>Oracle Warehouse Builder</i>
R-ETL	<i>Real-Time ETL</i>
RDBMS	<i>Relational Data Base Management System</i>
SAD	Sistema de Apoio à Decisão
SAP	<i>Systems Applications and Products</i>
SAP BW	<i>SAP Business Information Warehouse</i>
SAS	<i>Statistical Analysis Software</i>
SCD	<i>Slowly Changing Dimension</i>
SGBD	Sistema de Gestão de Base de Dados Relacionais
SSIS	SQL Server Integration Services
SLA	<i>Service-Level Agreement</i>

SQL	<i>Structured Query Language</i>
TOSDI	<i>Talend Open Studio for Data Integration</i>
XML	<i>eXtensible Markup Language</i>

Introdução

Com a crise que está por todo o mundo, as organizações também sofrem e alguns acabam por ficar por caminho.

Atualmente uma organização consegue elevar ou manter seu nível no mercado se for inovador, competitivo e flexível (segue as mudanças). No entanto, é necessário uma análise da empresa para saber os pontos mais altos ou mais baixos, por outras palavras, é conseguir ter uma visão transparente da saúde da empresa/organização.

A vida é feita de decisões que definem o nosso futuro, assim é também nas organizações em que há muitas decisões a serem tomadas. Essas decisões ditam o sucesso ou o fracasso de uma organização.

As decisões são tomadas a partir de uma análise, e essa análise é feita de acordo com os dados da organização, e esses dados são transformados e integrados para facilitar na análise, de modo que a decisão tomada seja a melhor possível.

Por conseguinte as organizações tendem a utilizar as novas tecnologias (BI) com o intuito de dar respostas de uma forma mais rápida e eficiente a esse mercado exigente e competitivo.

Segundo Misra & Rahman (2013), o papel das tecnologias de informação é cada vez mais importante para ajudar as organizações nas suas decisões. O crescimento das organizações, e o aumento da competitividade no mercado leva as empresas a procurarem as tecnologias de informações no sentido de dar respostas rápidas, com informações de qualidade e credível ao mercado exigente.

As organizações modernas estão adotando tecnologias de tomada de decisão como Data Warehousing, Data-Mining, Business Intelligence (Misra & Rahman, 2013). Da mesma forma afirma Ponniah (2010) dizendo que as empresas foram obrigadas a recorrer a novas formas de obtenção de informações estratégicas.

Segundo as análises de Ponniah (2010), podemos dizer que a informação é atualmente um dos bens mais importante de uma empresa, para poder ganhar ou manter a sua posição no mercado competitivo, e ser mais eficiente. De acordo com o mesmo autor, os executivos precisam dessas informações com conteúdo adequado para poderem tomar decisões estratégicas.

Mas para que a informação seja de qualidade é necessário um bom sistema de processamento dos dados. Segundo Hoffer *et al* (2010), a informação é um conjunto de dados que foram processados de modo que o conhecimento da pessoa que a usa é aumentada.

Se a informação é importante, os dados são quesitos importantes nas organizações e que aumentam a cada dia. Quando se fala de empresas do sector das telecomunicações, eles aumentam significativamente, é necessário um sistema capaz de integrar esses dados, que além de serem uma “ilha de dados”, provem de fontes heterogéneas, distribuídas e muitas vezes de sistemas legados.

Para melhor gerir esses dados as empresas têm apostado em sistema de apoio a decisão (SAD) baseada em arquiteturas DW (*Data Warehouse*). Contudo para a concepção de uma DW é necessário ter um processo de ETL (*Extract, Transform and Load*). Na perspetiva de Kimball & Caserta (2004, p.22), esse processo consome cerca de 70% dos recursos necessários para implementação e manutenção de um DW típico.

Segundo Kimball & Caserta (2004, p.22), um sistema de ETL vai além de extrair os dados do sistema fonte e colocar dentro de um DW. O presente trabalho visa ilustrar esses aspetos e mostrar que um processo ETL não é inerente ao DW. Podemos ter sistemas com destinos específicos que não seja DW.

Na mesma ótica o autor ainda refere que um sistema ETL:

- Remove os erros e corrige dados em falta;
- Fornece medidas documentadas de confiança nos dados;
- Captura o fluxo de dados transacionais por precaução;
- Ajusta os dados de várias fontes para ser utilizado em conjunto;
- Estrutura dados para serem utilizados por ferramentas de utilizadores finais.

Atualmente o mercado fornece diversas soluções para as empresas que querem fazer a integração de dados. Podemos encontrar soluções *open source* e comerciais. A escolha de soluções ETL não é fácil, depende das necessidades dos utilizadores finais. Há aqueles que preferem criar seu próprio sistema de integração de dados usando a codificação manual.

O trabalho tem um componente prático, onde é feito a análise de um processo de implementação de ETL, na empresa Unitel T+ Telecomunicações.

1 Objectivos

O presente trabalho tem com objectivo geral, compreender os principais mecanismos técnicos e ferramentas do processo ETL, bem como a sua implementação na Unitel T+. Para isso esse trabalho tem os seguintes objectivos específicos:

- Compreender os principais conceitos subjacentes ao processo ETL (*Extract, Transform and Load*);

- Analisar os aspectos mais relevantes das ferramentas ETL.
- Compreender o processo de implementação do processo ETL, os desafios encontrados, as soluções propostas, e os resultados obtidos.

2 Motivação

Dado a importância do processo ETL para as organizações, esse tema tornou-se muito interessante e que precisava de uma alavanca. É um tema que precisa de mais ênfase tendo em conta sua importância.

É um tema “novo”, e há poucos trabalhos científicos sobre esse assunto, apesar de haver bastantes artigos e livros sobre o mesmo.

ETL é uma das soluções de BI que vem ganhando terreno nas organizações no que toca o processamento dos dados.

Esse processo é muito utilizado na empresa Unitel T+ para integração de dados. Também ele é utilizado para auxiliar na realização dos relatórios e entre outros.

Neste sentido, pretende-se analisar o uso dessa tecnologia, na empresa Unitel T+, como sendo uma oportunidade de conhecimento e de novos desafios.

3 Metodologia

Para atingir esses objectivos, foi feito pesquisas bibliográficas relevantes, particularmente consultas de livros e *sites* na internet.

Também foram feitas entrevistas a algumas pessoas que tiveram contato com sistemas ETL na empresa Unitel T+, a fim de ter uma visão a nível do funcionamento do sistema. Mas também houve contribuição de outras pessoas, que foram importantes para o aumento do conhecimento a nível do uso das ferramentas.

Para uma melhor compreensão foi feita uma observação directa sobre o sistema que a Unitel T+ tem utilizado.

4 Estrutura do documento

O presente trabalho foi subdividido em 3 capítulos e se encontra estruturado da seguinte forma:

Na introdução começa-se pela contextualização, abordando também os objetivos (geral e específico), a motivação que levou a escolha do tema, a metodologia utilizada para alcançar os objetivos, e a limitação do estudo.

No primeiro capítulo fala-se do processo ETL, onde são definidos o conceito desse processo, a sua história, os requisitos de um ETL, as etapas que o compõem. Também fala-se um pouco sobre a integração dos dados, os subsistemas de ETL, e os metadados.

O segundo capítulo fala das ferramentas de ETL, onde são abordados o conceito dessas ferramentas, traçando a sua evolução, e descritas as que estão disponíveis no mercado. Ainda são enumerados alguns pontos que ajudam as organizações na escolha de uma ferramenta, ou no desenvolvimento de uma nova, usando a codificação manual. E por fim fala-se da ferramenta TOSDI.

No terceiro capítulo analisa-se um caso prático sobre um sistema ETL desenvolvido na empresa Unitel T+, e compreender a implementação desse sistema.

Por último a conclusão do presente trabalho, indicando os aspectos alcançados e os esperados com esse trabalho.

5 Limitações do estudo

O presente trabalho fala sobre o processo ETL, mas não como um subprocesso para concepção de um DW. Segundo Gama & Abreu (2008), muitas vezes a ideia que se tem é que o processo ETL é inseparável ao DW, porque, esse é muitas vezes ensinado como

sendo um subprocesso no processo de construção de um DW ou Data Mart. Também nesse trabalho fala-se sobre as ferramentas ETL para integração de dados, por isso as ferramentas de análise e consulta não serão abordados.

Capítulo 1: Processo ETL

1 Introdução

Neste capítulo vamos falar sobre o processo ETL, em que consiste esse processo, quais as etapas mais e menos importantes, os requisitos de ETL, os subsistemas desse processo e um pouco da sua história. Também falaremos sobre a integração de dados, e os metadados de ETL.

2 Conceito de ETL

Segundo Ribeiro (2011), a *Extract Transform Load* (ETL), refere a **Extracção** de dados de uma ou várias fontes diferentes, **Transformá-los** e **Load** desses dados para um ou vários repositórios finais (DW ou BD específico, ou algum ficheiro de arquivo).

Um sistema ETL é composto por três (3) etapas ou fases (Kozielski & Wrembel, 2008, p. 21):

1. **Extracção** – esta é a primeira fase onde faz-se a extracção dos dados nas fontes heterogéneas.
2. **Transformação** – nesta fase ocorre a transformação dos dados, ou seja, a limpeza, a padronização do formato dos arquivos, etc.

3. **Load de dados** – nesta fase os dados transformados são inseridos em um ou vários BD de destino (DW/Data Mart), ou para um repositório final específico.

A figura a seguir ilustra a arquitectura geral de um processo ETL, onde o primeiro bloco representa as fontes de dados (*e.g. Flat files, Base de Dados, ficheiros XML, Excel, etc.*), o segundo representa a transformação desses dados, e por último a inserção (*Load*) dos dados num repositório de destino.

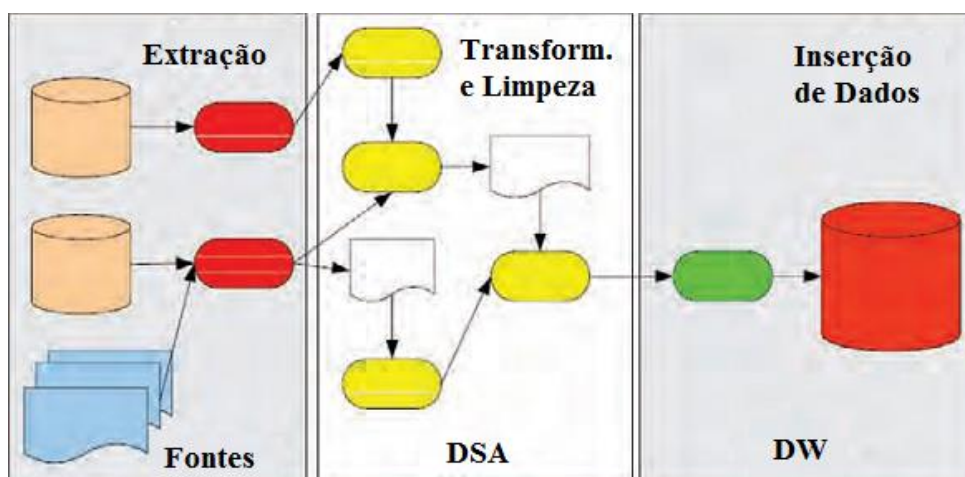


Figura 1 - Ambiente do processo ETL

Fonte: Adaptado (Wang, 2008)

Segundo Gama & Abreu (2008), as fases de extracção e inserção de dados são obrigatórias, sendo a fase de transformação opcional, isso porque, podemos pegar dados de uma (ou várias fontes) e fazer somente a migração desses dados, sem mesmo ter que modificá-los.

Numa definição mais aprofundada, Kimball & Caserta (2004, p. 462) dizem que essas são as três (3) fases clássicas do processo ETL, e que pode ser expandido em quatro (4): extracção, limpeza, confirmação, e entrega.

Na próxima seção vamos falar mais aprofundado sobre essas etapas que compõem o processo ETL.

Ribeiro (2011) diz que um sistema ETL tem que suportar dados de diferentes tipos, comunicar com base de dados distintos e ler diversos formatos de arquivos.

Para Gama & Abreu (2008), o ETL é vulgarmente conhecido como um subprocesso do processo para construção de uma DW, porque se encontra associado aos projectos de DW, mas isso não significa que não podemos ter um sistema ETL para outros sistemas específicos.

Esse processo é o mais crítico, complexo e o que consome mais tempo num projecto de DW e segundo Kimball & Caserta (2004, p.22), esse pode consumir de 70% dos recursos necessários para a implementação de um projeto de DW. Porque todo esse processo, desde a colecta de dados, passando pela transformação (o que consome mais tempo ainda), até a inserção desses dados, é duradouro, complexo e crítico.

Não vale a pena passar por todo esse processo e no final os dados não forem confiáveis, ou limpos, por esta razão que esse processo é muito importante no que diz respeito a inteligência de negócio.

Atualmente as empresas que optarem por usar sistemas ETL têm várias formas de o fazer, isto porque podem optar pelo desenvolvimento manual ou por uso das ferramentas existentes no mercado (comerciais ou *OpenSource*).

Na perspectiva de Neto (2012, p. 13), desenvolvimento manual apesar de responder exactamente a demanda da instituição (Cliente/Empresa), os desenvolvedores enfrentam uma maratona de dificuldades, e o projecto pode estar susceptível a erros, demorar bastante tempo, e a qualidade pode estar comprometida.

Além disso, o autor ainda diz que é necessário ter uma equipa com elementos inteligentes e programadores especializados.

Portanto a decisão da escolha entre uma e outra forma na construção de um sistema ETL deve ser bem cuidada.

No caso do uso das ferramentas de ETL, esse assunto vamos falar no Capítulo 2.

3 História de ETL

De acordo com Hammergren & Simon (2009), foi à muito tempo atrás nos finais do Séc. XIX e no início do Séc. XX, quando *Charles Coolidge Parlin* e *Arthur C. Nielsen* ambos pesquisadores de mercado analisavam alta qualidade de dados para poder ajudar as empresas nas tomadas de decisões.

Os mesmos autores ainda dizem que nos anos 70's mesmo com o domínio dos computadores de grande porte, onde os dados passaram a ser organizados e categorizados, esses eram armazenados dentro dos ficheiros ou base de dados separadamente, e era impossível fazer o cruzamento desses dados, para uma melhor análise da empresa. Para tentar resolver esse problema foi criada uma empresa chamada de *Teradata*, no ano de 1979, que usava processamento paralelo com múltiplos microprocessadores para gerir *terabytes* de dados.

Ainda segundo Hammergren & Simon (2009), no ano de 1980, com o crescimento dos computadores nas empresas, os dados aumentaram, mas ainda encontravam isolados em ficheiros de arquivos e BD. Como esse problema persistia, um grupo de empreendedores pensaram em criar um *software* onde podia responder várias questões usando o cruzamento dos dados, e esse software chamava-se *Distributed Database Management System* (DBMS distribuído, ou simplesmente DDBMS) (em Português, **Sistema de Gerenciamento de Base de Dados Distribuídos (SGBD)**). Mas ainda o problema de “Ilhas de dados” continuava, até que *Teradata* começou a resolver esses problemas. O primeiro sistema *Parallel Relational DBMS* (RDBMS Paralelo) de teste foi recebido por *Fargo Bank* em 1983. E em 1984 lançou a sua primeira versão do sistema.

Dáí outras empresas começaram a criar os seus próprios sistemas e, segundo Hammergren & Simon (2009), em 1986 Ralph Kimball fundou *Red Brick Systems*, onde usavam *Indexes* para melhorar performance nas consultas grandes e complexas sobre uma plataforma de BD relacionais.

Segundo Inmon (2008), os primeiros a adotaram o DW foram as Telecomunicações, Companhias de Seguros, Bancos e retalhistas. Depois de algum tempo *Data Warehousing* se espalhou para outras empresas.

Daí muitas empresas começaram a desenvolver ferramentas para ajudar outras empresas na construção de DW.

De acordo com Hammergren & Simon (2009), em 1993, Inmon escreveu o livro *Building the Data Warehouse* (Wiley), por isso ele é considerado, por muitos, como pai da DW. E em 1996 Ralph Kimball, lançou também o livro sobre DW intitulado de *The Data Warehouse Toolkit* (Wiley). Daquele tempo até os nossos dias começaram a ser publicados muitos livros, artigos e mais sobre esse assunto, complementa.

Mas só no ano 1990 apareceu as primeiras formas de utilização de ETL, onde utilizavam a codificação manual para desenvolvimento de um sistema de ETL, segundo a tabela a seguir.

Ano	Título	Significado
Início de 1990	Codificação manual de ETL	Códigos personalizados escritos à mão
1993-1997	A primeira geração de ferramentas de ETL	Código baseado em ferramentas de ETL
1999-2001	Segunda geração de ferramentas de ETL	Código baseado em ferramentas de ETL
2003-2010	Ferramentas de ETL atualmente	A maioria das ferramentas eficientes

Tabela 1 - As várias gerações de ETL ao longo dos anos.

Fonte: Adaptado (Ferreira, 2010)

Segundo Inmon (2008), a partir de 1995 a expansão de ETL no mercado foi rápida, e despertou interesse de muitas empresas.

(...) outras empresas começaram a produzir suas próprias versões de ETL. Microsoft produziu um produto representada como um produto de ETL. Esse produto foi nomeado de DTS. SAP produziu sua própria versão de ETL, e Oracle produziu sua versão de ETL. Os produtos SAP, Oracle e Microsoft não foram projetados para uso geral do ETL no sentido de que a plataforma de destino não era ilimitado. Por exemplo, ETL SAP só produz saída com destino a SAP BW. (Inmon, 2008).

Na perspectiva de Inmon (2008), o custo de um sistema de ETL era muito elevado, muitas empresas pequenas e de médio porte não conseguiriam suportar esse custo, por isso que apareceu no ano de 2000, uma nova solução, a introdução de um novo tipo de ETL para mercado de médio porte. O autor disse ainda que o Talend era o líder do mercado nesse novo espaço, pois “ (...) se encaixa no mercado com boa funcionalidade a um preço significativamente inferior ao de qualquer outro concorrente”. (Inmon, 2008).

Mas com a evolução do ETL, muitas características foram adicionadas ao longo desse percurso, como por exemplo, “ (...) paralelismo, captura automática de metadados, forma livre codificação, e output para Java e outras linguagens.” (Inmon, 2008).

Hoje podemos contar com centenas de *softwares* que trabalham com integração de dados, e de entre eles, na perspectiva de Thoo *et al.* (2012)¹, os que destacam no mercado atualmente são:

- A Informática;
- A IBM;
- O SAP;
- O Oracle; e
- SAS-DataFlux.

¹ A última pesquisa feita por Gartner sobre *Ferramentas de Integração de Dados* em Outubro de 2012

Em 2008 Inmon disse que o futuro dos sistemas de ETL era para os dados não estruturados, e presentemente (2013), podemos usufruir de muitas soluções (comercial e *OpenSource*) no mercado que suporta essa funcionalidade, de entre eles o *Talend Big Data*, mas não vamos aprofundar muito sobre esse assunto.

Sobre as ferramentas de Integração de Dados (de ETL) vamos falar no Capítulo 2.

4 Requisitos de um ETL

Antes de começar um projeto de ETL temos que ter uma estratégia e definir alguns requisitos essenciais. “Certifique-se de saber suas necessidades antes de começar o ETL.” (Ross & Kimball, 2004).

De acordo com Kimball & Caserta (2004), nós começamos o projeto de ETL com um dos mais difíceis desafios (os requisitos/necessidades), ou seja, reunir num só lugar todos requisitos conhecidos, realidades e limitações que podem afetar o sistema de ETL.

“Seguindo os requisitos, identificamos uma série de decisões de arquitetura que você precisa fazer no início do seu projeto de ETL.” (Kimball & Caserta, 2004, p. 38). Segundo o mesmo autor, essas decisões são importantes porque ela guia tudo o que fazemos enquanto avançamos para a implementação.

Segundo Ribeiro (2011), os seguintes itens tem que estar bem alinhados antes de qualquer projeto de ETL:

- **Requisitos de Negócio** - Você tem bem claro e documentado quais são os requisitos de negócio?
- **Viabilidade dos Dados** - Foi realizado uma análise da viabilidade dos dados?
- **Latência dos Dados** - Qual é o tempo máximo permitido para disponibilização dos dados através do sistema de BI?

- **Políticas de conformidade e segurança** - Quais são as políticas de conformidade e segurança adotadas pela empresa?

Outros autores escreveram sobre os requisitos de ETL, entre eles se destaca Kimball, porque ele fala de uma forma mais abrangente sobre esses requisitos, e segundo ele os seguintes requisitos podem ser fundamentais para um projeto ETL (Kimball & Caserta, 2004):

- **Necessidades do Negócios**
- **Requisitos de Conformidade**
- **Perfil de Dados**
- **Requisitos de Segurança**
- **Integração de Dados**
- **Latência de dados**
- **Arquivamento e Linhagem**
- **Interfaces de utilizador final**
- **Habilidades disponíveis**
- **Licenças legadas**

Para uma maior compreensão recomendamos a leitura desses requisitos no livro (Kimball & Caserta, The data warehouse ETL toolkit : practical techniques for extracting, cleaning, conforming, and, 2004, p. 39).

5 As etapas do Processo ETL

Como não existe um processo sem suas etapas, eis que vamos falar, nessa seção sobre as etapas que compõem o Processo ETL.

Conforme referido anteriormente ele é composto por 3 etapas básicas, a **Extracção (E)** dos dados de uma ou várias fontes heterogêneas, a **Transformação (T)** desses dados e por último, o **Load (L)** desses dados num repositório de destino (que pode ser um ou vários).

A figura a seguir mostra o ciclo de um ETL.

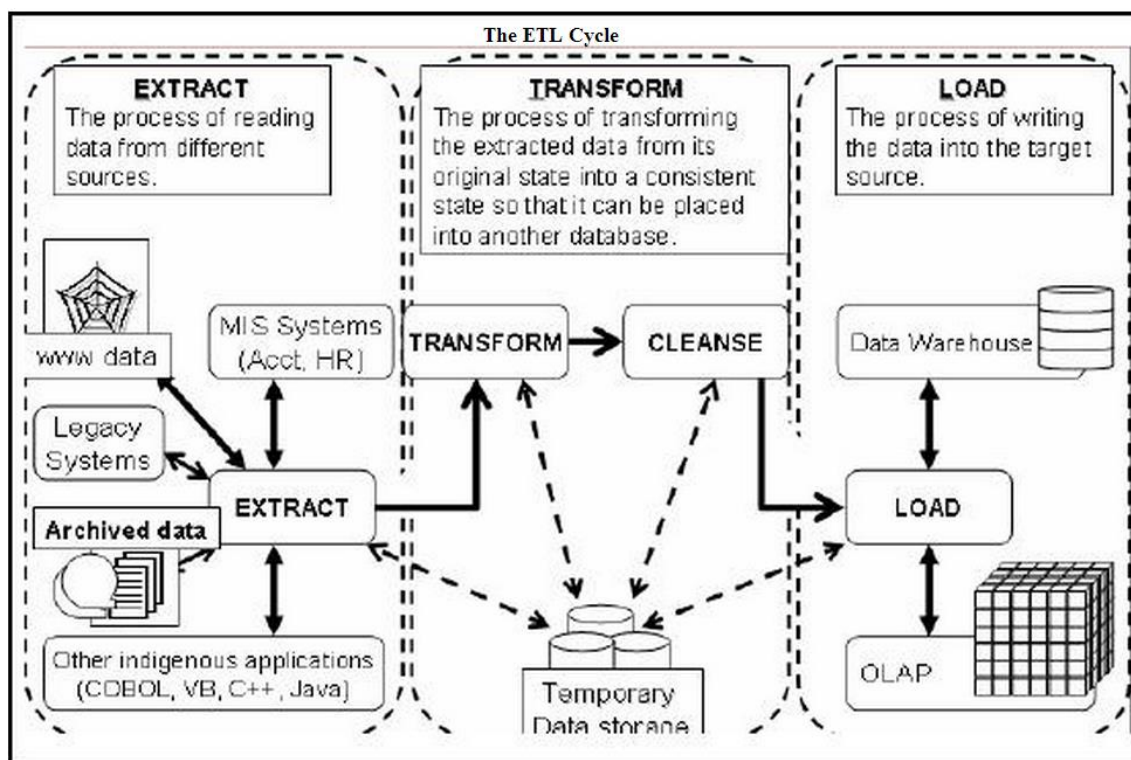


Figura 2 - O ciclo de ETL²

Vamos começar por falar da primeira etapa que é a extracção de dados.

² Disponível em:

http://www.zeeopedia.com/read.php?extract_transform_load_etl_etl_cycle_processing_data_extraction_data_transformation_data_warehousing&b=6&c=16, consultado a 24 de Junho de 2013

5.1 A Extração de dados

“Durante o processo de ETL, a primeira tarefa que deve ser realizada é a extracção da informação relevante que tem ainda de ser propagada para o *warehouse*”. Theodoratos et al. (2001) *apud* Wang (2008, p. 574).

Para Liu & Özsu (2009, p. 1096), a etapa de extração é conceitualmente a tarefa mais simples de todos, com o objetivo de identificar o subconjunto correto de dados de origem que tem de ser submetido ao fluxo de trabalho de ETL para processamento posterior. Mas para Lane (2005, p. 173), projetar e criar o processo de extração é muitas vezes um dos mais demorado tarefas no processo de ETL e, de fato, em todo o processo de *DW*.

Pelo que podemos constatar desses dois autores, a extração de dados é a “mais simples”, no entanto, isso não significa que não demora muito tempo.

Na perspectiva de Hoffer *et al.* (2010), a captura de dados relevantes em fontes como, ficheiros e BD para preencher *EDW* é chamado de *Extração*. Dizem ainda que, normalmente nem todos os dados contidos nas fontes são importantes, e que para extrair esses conjuntos de dados importantes, é necessária uma análise extensa de ambas as partes (tanto para a fonte de dados como para o sistema final).

Na mesma linha Kimball *et al.* (2008, p. 127), diz que “na maioria das vezes, o desafio no processo de extração é determinar quais dados para extrair e que tipos de filtros aplicar”.

Segundo Wang (2008, p. 574), essa etapa envolve somente uma fração dos dados alterados desde a ultima execução do Processo ETL, com o objectivo de minimizar o tempo de processamento geral. Mas isso caso não for a primeira extração.

Podemos ver que a extração de dados nunca pode extrair dados que já foram processados com sucesso, salvo se houver uma actualização, eis a razão pelo qual a extração deve ser capaz de detetar os dados actualizados.

De acordo com Hoffer *et al.* (2010), temos dois tipos genéricos de extração de dados, a extração estática e a incremental. Onde a extração estática é usada inicialmente para preencher o *DW*, e a extração incremental é utilizada para manutenção do *DW*.

- A **extração estática** é um método para capturar *snapshots* dos dados de origem exigidos num determinado período de tempo.
- A **extração incremental** captura apenas as mudanças que ocorreram nos dados de origem desde a última captura, um exemplo comum, é a captura de *logs*.

Mas ainda existe outros métodos de extração de dados para *DW*, o método de extração lógica e o de extração física (Lane, 2005).

5.1.1 Método de extração Lógica

De acordo com Lane (2005), temos dois tipos de extração lógica, a extração total e a extração incremental:

- **Extração Total**

De acordo com Lane (2005, p. 174), neste tipo de extração os dados são extraídos por completo, porque não há controlo das alterações que houveram com os dados nas fontes, desde a última extração com sucesso.

Portanto não é necessário saber se houve ou não alteração nos dados de origem, por isso pode consumir mais ou menos tempo, dependendo da quantidade dos dados no sistema de origem e dos que já estiverem no destino.

Tableau (2012) complementa dizendo que “embora este tipo de atualização garante que você tenha uma cópia exata do que está na origem dos dados, às vezes pode levar muito tempo e ser dispendioso na base de dados dependendo de quão grande é o extrato.”

- **Extração incremental**

Essa extração captura apenas as mudanças que ocorreram nos dados de origem desde a última captura (como tinha referido anteriormente). Segundo Lane (2005), para identificar essa mudança, deve haver a possibilidade de identificar todas as informações alterado desde a última captura, e essa informação pode ser fornecida por uma coluna nos dados de origens, identificando a última hora que foi feita a alteração, ou uma tabela de alteração que mantém as informações das mudanças.

5.1.2 Método de extração Física

Os dados podem ser fisicamente extraídos por dois mecanismos, obedecendo o método de extração lógica escolhida (Lane, 2005). E segundo mesmo autor, esses mecanismos podem ser, extração Online a partir do sistema de origem ou a partir de uma estrutura Offline que pode já existir ou pode ser gerado por uma rotina de extração.

- **Extração Online**

Aqui os dados são extraídos diretamente da fonte para processamento na *Staging Area*, é por isso que ele é chamado de extração *online* (K, 2013). Na mesma perspectiva Lane (2005) diz que os dados são extraídos diretamente do sistema fonte.

- **Extração Offline**

Segundo K (2013) e Lane (2005), nesse tipo de extração os dados não são extraídos directamente da fonte, em vez disso são obtidos a partir de uma outra zona externa, que mantém a cópia dos dados de origem. Aqui um sistema intermediário não é necessário.

Essa fonte externa pode ser (Lane, 2005):

- *Flat files*;
- *Dump files*;
- *Logs*;

- Etc.

De acordo com Liu & Özsu (2009), essa etapa tal como os outros, ela precisa ocorrer num período especial (e.g. à noite seria um período ideal). Ainda segundo Liu & Özsu (2009, p. 1096), essa etapa tem uma considerável dificuldade devido a duas restrições técnicas:

- A fonte de dados deve sofrer o mínimo de sobrecarga durante a extração, já que outras atividades administrativas também acontecem durante esse período.
- Deve haver um mínimo de interferência com a configuração do *software* no lado da fonte de dados porque, tanto por razões técnicas e políticas, os administradores não aceitam grandes intervenções para a configuração de seu sistema.

5.2 Transformação e Limpeza de dados

Esta é a segunda etapa do processo ETL, onde os dados passam por uma transformação, ou seja, os dados vão ser padronizados conforme as necessidades requeridas. “As tarefas de transformação e limpeza constituem a principal funcionalidade de um Processo ETL.” (Kozielski & Wrembel, 2008).

Para Ponniah (2010), os dados tem que ser útil no *DW*, e são extraídos de várias fontes e muita das vezes de sistemas legados, e a qualidade desses dados provavelmente não é boa suficiente para ser inserida no *DW*.

Por esta razão é necessário temos uma etapa onde esses dados passam por uma reconciliação/padronização antes de serem inseridos no *DW*. De acordo com as ideias do autor podemos dizer que uma boa fonte de dados exige pouca transformação, por isso, a quantidade de manipulação necessária depende das fontes de dados.

Os dados extraídos passam por uma área intermediária de armazenamento, chamado de DSA (Kozielski & Wrembel, 2008).

“Depois que os dados forem extraídos, serão transportados para um sistema de destino ou para um sistema intermediário onde passam por um processamento adicional. E dependendo do tipo de transporte escolhido, algumas transformações podem ser feitas durante o processo.” (Lane, 2005).

Nem sempre um processo ETL passa por essa etapa, porque as vezes pode-se querer somente uma migração de dados (transportar dados de uma origem para um destino específico).

Também existe os dados chamados de “limpos” em que não será necessário haver uma transformação.

De acordo com Hoffer *et al.* (2010), a transformação dos dados está no centro do processo de reconciliação, e ela envolve a conversão dos dados no formato da fonte, para o formato do sistema de destino, que pode ser *DW/Data Mart*, ou um sistema de destino específico. Essa etapa envolve uma aplicação com uma série de regras ou funções³ para os dados extraídos.

A transformação dos dados é muito importante e complexo por isso, podemos encontrar alguns autores que preferem falar desse assunto em partes como *Cleaning e Conforming* (Kimball & Caserta, 2004) ou *Transformation & Cleaning* (Wang, 2008), e em português, alguns preferem, Limpeza, Ajustes e Consolidação (Ribeiro, 2011), mas todos formam a etapa de transformação de dados.

Segundo Liu & Özsu (2009) e Kozielski & Wrembel (2008), as tarefas de transformação e limpeza lidam com classes de conflitos e problemas que são categorizados da seguinte forma:

- **Problemas ao nível de *Schema*** - os principais problemas em relação ao nível do *schema* são: (a) os conflitos de nomenclatura, onde o mesmo nome é usado para objetos diferentes (homónimos) ou nomes diferentes são usados para o mesmo objeto (sinónimos), e (b) os conflitos estruturais, onde deve lidar com

³ Para mais informações visite <http://www.etltools.org/>

diferentes representações do mesmo objeto em diferentes fontes, ou converter tipos de dados entre fontes e o *DW*.

- **Problemas ao nível de Registo** - os problemas mais comuns em relação ao registo, são registos duplicados ou contraditórios, e problemas de consistência.
- **Problemas ao nível de Valores** - aqui se destacam, problemas de baixo nível técnico, tais como representações de valores diferentes ou interpretação diferente dos valores. Como exemplos, diferentes representações de valores (e.g., para o sexo: “Homem”, “M” ou “1”), ou diferentes interpretações de valores (e.g., formatos de data: Americano “mm/dd/yy” vs. Europeu “dd/mm/yy”).

5.2.1 Funções da transformação de dados

Como já referimos anteriormente nessa etapa é usada várias funções para resolução dos vários problemas supracitados.

Para lidar com essas questões, as tarefas de integração e transformação envolve uma grande variedade de funções, como a normalização, desnormalizar, reformatar, recalcular, resumir, a fusão de dados de múltiplas fontes, modificando estruturas-chave, adicionando um elemento de tempo, identificando os valores padrão, fornecendo comandos de decisão para escolher entre várias fontes, e assim por diante. Adaptado (Kozielski & Wrembel, 2008, p. 23).

Segundo Tools (*cf.2011*), durante essa etapa é comum a utilização de alguns processos como, conversão, limpeza de duplicados, padronização, filtragem, ordenar/classificar, *Match*, e entre outros, para que os dados estejam em conformidade, e passem para a próxima etapa onde vão ser inseridos num sistema de destino.

Transformação de dados engloba variedade de funções e na perspectiva de Hoffer *et al.* (2010), podem ser classificados “brutalmente” em duas categorias, funções ao nível de registo e funções ao nível de campo (*field-level*), em que:

- **Funções ao nível de registo**

As funções mais importantes para este nível, na perspectiva de Hoffer *et al.* (2010), são a seleção, junção, normalização, e agregação.

A **Seleção**, é um processo onde os dados são particionados de acordo com os critérios preferidos.

A **Junção**, como o próprio nome diz, ela faz a junção de dados de várias fontes.

A **Normalização** é o processo de decomposição de relações com anomalias para produzir relações menores, bem estruturados.

A **Agregação** é o processo de transformar dados de um nível detalhado para um nível de resumo.

- **Funções ao nível de campo**

Segundo Hoffer *et al.* (2010), essa função converte os dados de um determinado formato em um registo de fonte para um formato diferente no registo alvo. Nesse nível temos dois tipos de funções: funções de campo único e funções de diversos campos.

- **Função de campo único** - converte dados de um campo de fonte único para um campo de destino único. A figura a seguir ilustra uma representação básica desse tipo de função, onde há uma transformação (T) para um determinado campo.

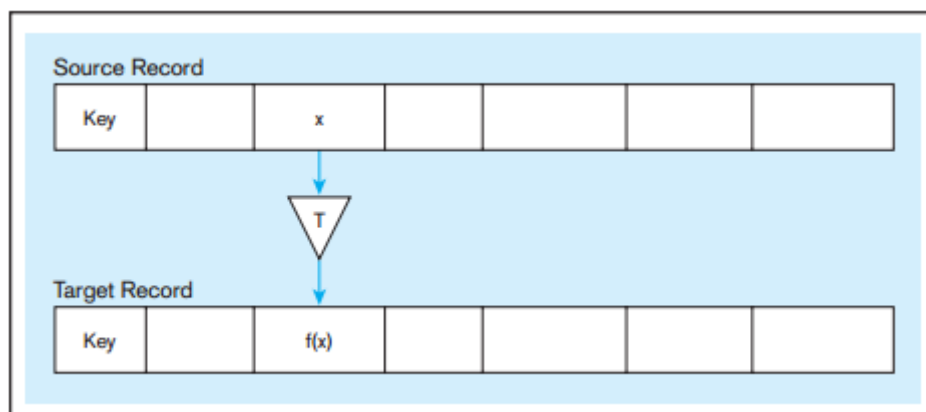


Figura 3 - Representação básica para transformação de campo único

Fonte: Hoffer *et al.*(2010, p.454)

- **Função de diversos campos** - converte dados de um ou mais campos de origem para um ou mais campos de destino. Este tipo de transformação é comum nos DW. A figura a seguir ilustra dois exemplos desse tipo de função, em que a) representa muitos para um e b) representa um para muitos.

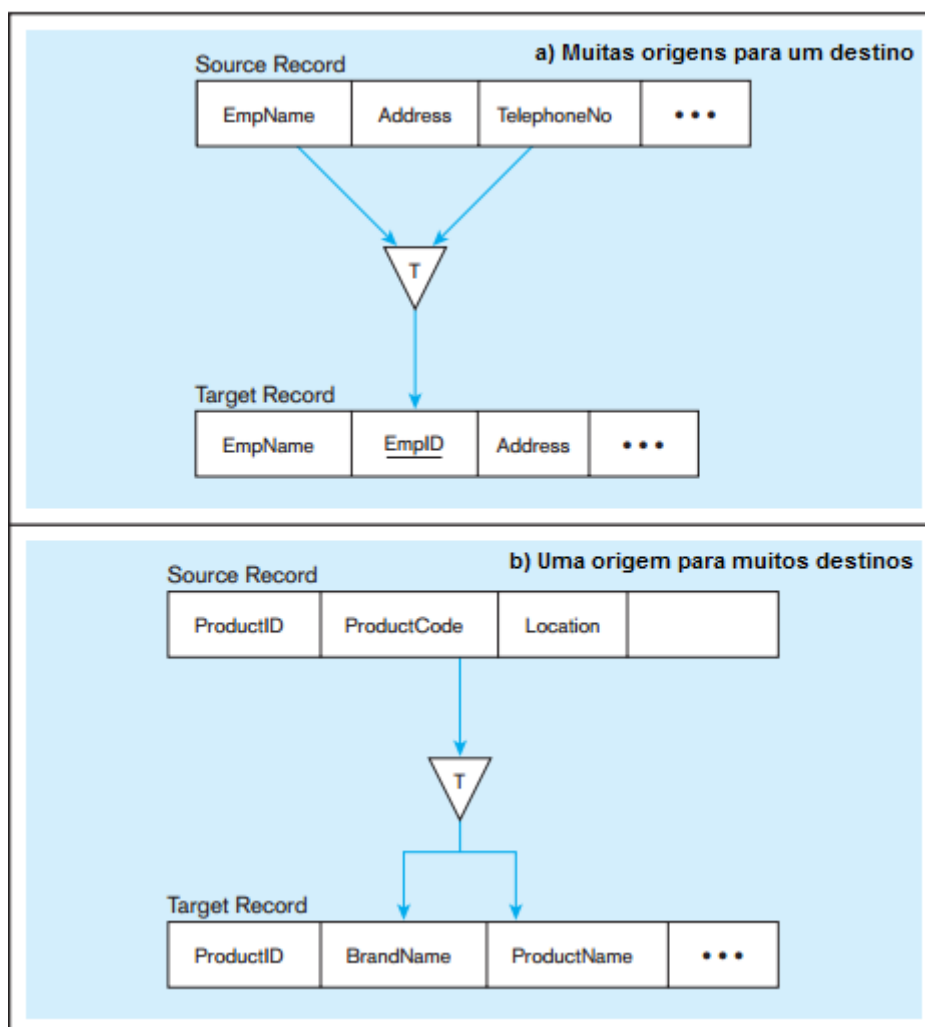


Figura 4 - Transformação de diversos campos

Fonte: Adaptado (Hoffer *et al.*,2010, p.455)

5.3 Load ou Inserção de dados

A fase final do *workflow* de ETL vem com a entrega dos dados para as tabelas apropriadas (Liu & Özsu, 2009). Nessa etapa os dados já se encontram extraídos e limpos, prontos para serem carregados para um sistema final, por isso ela é a última fase do Processo ETL, como mostra a Figura 5 - A última etapa do Processo ETL.

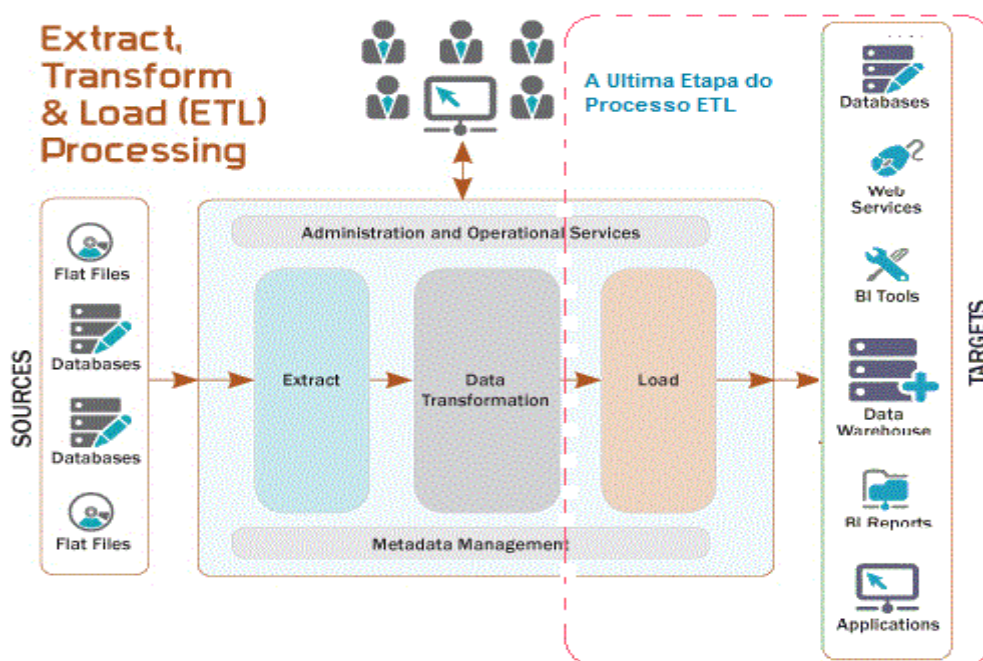


Figura 5 - A última etapa do Processo ETL

Fonte: Adaptado (Sherman, 2009)

Como vimos na seção anterior os dados podem ser extraídos de uma ou várias fontes, e nessa etapa também os dados podem ser carregados para um ou vários repositórios finais.

Segundo Ponniah (2010), carregamento de dados pode consumir uma enorme quantidade de tempo, por isso essa etapa é geralmente causa de grande preocupação, e não devemos pensar que durante um processo de *Load*, os dados serão carregados com sucesso para um sistema de destino. Por isso que temos que ter um sistema capaz de detectar esses dados que não foram inseridos com sucesso e reportar em log ou em outra forma mais conveniente.

Sempre que terminar o *Load* de dados temos que verificar se esses dados estão conforme os requisitos pretendidos, ou criar uma lógica capaz de o fazer, mas mesmo assim seria melhor a primeira opção (Ou as duas).

Quando fazemos o *Load* de dados, as vezes, esses dados se encontram em sistemas diferentes do sistema de destino, por isso, Ponniah (2010), nos diz que devemos considerar o impacto que o transporte desses dados pode causar na rede. Muitas das vezes se a aplicação não for eficiente, podemos ter uma sobrecarga no *Load* de dados, portanto temos que programar/agendar um tempo específico para essa operação.

Para amenizar essa situação, podemos utilizar a compressão de dados e sempre ter um plano de contingência (Ponniah, 2010).

Para uma melhor tomada de decisão nos negócios, as vezes é necessário que tenhamos os dados mais atualizados possível, ou seja dados em “tempo-real”, mas nem sempre uma melhor decisão necessita de dados actualizados em tempo real, por isso temos uma frequência com que os dados podem ser carregados. Essa frequência pode ser uma hora, um dia, ou ainda um mês. Cada um agenda/programa de acordo com a sua necessidade de negócio.

Segundo Ponniah (2010), temos 3 tipos de *Load* de dados:

- ***Load Inicial***

Esse tipo de *load* de dados, ocorre quando vamos inserir os dados no sistema de destino pela primeira vez.

- ***Load Incremental***

Quando que os dados do destino não estiverem actualizados, para que haja uma actualização de somente os dados em falta, usa-se esse tipo de *load* de dados, porque esse utiliza uma forma de saber a última vez que os dados foram inseridos. Por isso somente os dados alterados depois dessa data serão carregados.

- **Full Load**

Os dados que se encontram no sistema de destino serão apagados (*Truncate* se for necessário) e carregados os que foram extraídos para esse destino. É como carregar todo o sistema de origem para dentro do sistema de destino, mas é importante não esquecermos que podemos ter várias fontes de dados.

Quando fazemos o *load* de dados pela primeira vez é a mesma coisa que o *load* total (*Full Load*), pois no sistema de destino não se encontra nenhum registo. É bom não esquecer que quando nos referimos a “Sistema de Destino”, pode ser um ou mais Sistemas (pode ser DW/DM ou sistema específico).

Para K (2012) esse tipo de *load* consome mais tempo e é a mais simples.

❖ Comparação entre Incremental e Full Load

Incremental *load* consome menos tempo que o *load* total e é a mais difícil, porque temos que detetar as alterações e fazer atualização desse dados (K, 2012). A Tabela 2 mostra a comparação entre esses dois tipos de *load* de dados.

Full Load	Load Incremental
<i>Truncate</i> todas as linhas e cargas de dados a partir do zero.	Actualização de cada novo registo.
Requer mais tempo.	Requer menos tempo.
É mais fácil.	Difícil. ETL deve verificar se há novas linhas ou linhas atualizados.
Pode ser perdido.	Pode ser preservado/mantido

Tabela 2 - Comparação Incremental e Full Load.

Fonte: Adaptado (K, 2012)

6 Integração de Dados

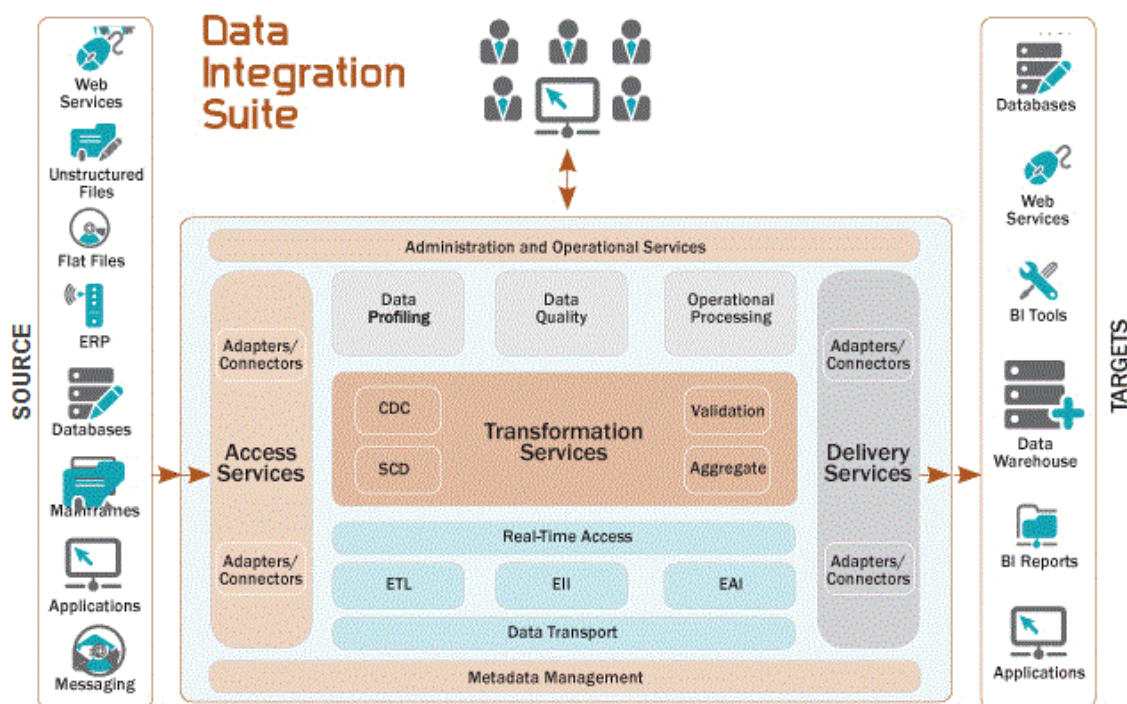
Nesta seção vamos falar um pouco sobre a integração de dados e seus antecedentes históricos. De acordo com Hoffer *et al.* (2010), é preciso saber alguns conceitos sobre a integração de dados, uma vez que trabalhamos com BD criados por outras fontes existentes.

Muitas vezes as empresas têm seus dados em fontes e locais distintos (dados internos e/ou externos) (Hoffer *et al.*, 2010), mas esses dados sozinhos não ajudam para uma melhor tomada de decisão, então, é preciso que haja uma conexão/ligação entre esses dados, é aí que entra a integração de dados.

A Figura 6 ilustra as diferentes fontes no qual podemos encontrar, um conjunto de integração dos dados e os possíveis destinos.

Segundo Liu & Özsu (2009), essa integração de dados permite **grandes consultas e análises nunca antes possíveis** com o uso das fontes de dados de forma individuais, tudo isso devido a **combinação desses dados** provenientes dessas fontes distintas.

O núcleo para qualquer método de integração de dados são as técnicas para capturar dados alterados, CDC, onde captura somente os dados alterados desde a última actividade de integração de dados, para uma possível atualização, isso pode ser feito usando *flags*, ou a data da última atualização (Hoffer *et al.*, 2010, p. 444).

Figura 6 - Integração de Dados⁴

6.1 Enquadramento Histórico

Segundo Liu & Özsu (2009), quando uma aplicação precisa de integrar os dados de fontes distintas, a solução poderá ser a codificação de algumas funcionalidades como a transformação e agregação, e incluir essas funcionalidades na aplicação.

Porém, isto traz algumas complicações, pois é um processo complexo e demorado, e que pode afetar a robustez e a capacidade de manter a aplicação. Essa solução não era muito viável, pois, sempre que precisasse fazer um cruzamento dos dados, teríamos que codificar. Se tivermos um *Terabyte* (1TB) de dados para processar, teríamos que ter uma solução que fosse mais eficiente, pois essa solução não seria a melhor.

Isso levou ao “ (...) desenvolvimento de arquiteturas e metodologias que abstraem a funcionalidade de transformação e agregação de dados em *software* de integração de dados genérico.” (Liu & Özsu, 2009).

⁴ Fonte disponível em:

http://gerardnico.com/wiki/detail/dw/etl/data_integration_suite.gif?id=dit%3Aetl_become_di, consultado a 20-04-2013

Com o aumento da variedade, complexidade, volume de dados, a integração de dados tornou-se um quesito imperativo, por isso, foram desenvolvidas arquiteturas e sistemas para o suportar (Liu & Özsu, 2009).

Desde os inícios do ano de 1990 até hoje (2013), muita coisa foi feita no que toca ao desenvolvimento e metodologias para integração de dados (Liu & Özsu, 2009). Desde então, muitas ferramentas foram desenvolvidas e muitas funcionalidades apareceram.

No [Capítulo 2](#) falaremos sobre essas ferramentas de integração de dados.

A integração de múltiplas fontes de dados heterogenias é atualmente como uma base para o sucesso das empresas, pois ela vem ajudando muito no que tange à análise dos dados em diferentes fontes.

Ela vem preocupando muito as grandes empresas, porque na sua falta, isso implicará um esforço maior por parte das equipas técnicas, demora muito tempo para um relatório ficar pronto, por outras palavras, isso pode levar a perda de oportunidades.

Tendo em conta Searls (2003), hoje as empresas querem cada vez mais ser mais competitivas, por isso, elas estão a adotar essa “nova” ideia, para ganharem tempo e qualidade nos seus serviços.

6.2 Abordagem para Integração de Dados

Existe algumas abordagens gerais que são muito úteis e podem ser utilizados para a integração de dados, cada um com um propósito diferente e cada um sobre circunstâncias distintas (Hoffer *et al.*, p. 444). Segundo eles temos as seguintes abordagens para integração de dados:

- Consolidação de Dados;
- Federação de Dados;

- Propagação de Dados.

6.2.1 *Consolidação de Dados*

Nessa técnica os dados são capturados de muitas origens para passar por um tratamento e serem integrados num repositório único (Josko, s.d.). Aqui os dados passam por uma transformação, como o próprio nome diz, uma consolidação.

Na perspectiva de Josko (s.d.), essa é a principal técnica na concepção de uma DW. A transformação de dados inclui, a reestruturação, limpeza, agregação, reconciliação, e entre outros.

Na mesma linha Josko (s.d.), afirma que a tecnologia de ETL utiliza essa técnica para consolidação dos dados estruturados e a tecnologia de ECM foca no trato de dados não estruturados como documentos diversos e páginas Web. A figura mostra a arquitetura da consolidação de dados.

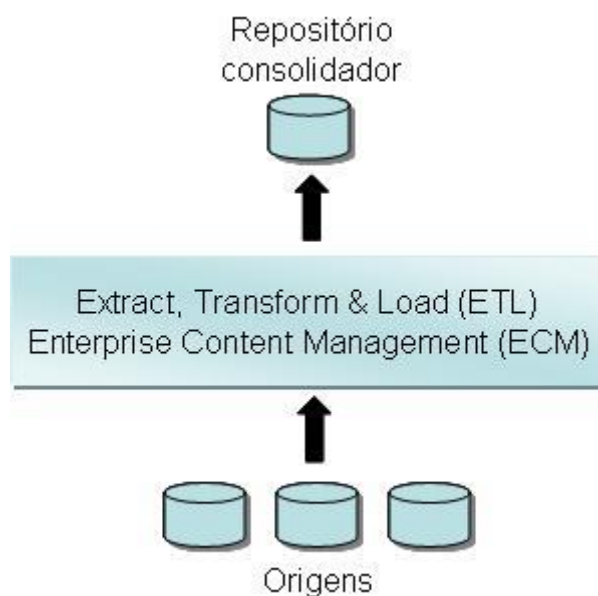


Figura 7 - Arquitetura de Consolidação dos Dados e as Tecnologias que a apoiam

Fonte: (Josko, s.d.)

Esse trabalho é baseado nessa técnica, e não será necessário falarmos muito aqui.

6.2.2 Federação de Dados

Essa é uma técnica para integração de dados, também conhecido como **Virtualização de Dados**, que fornece uma visão virtual de dados integrados, sem realmente criar uma BD centralizado (Hoffer *et al.*, 2010, p. 444). Dizem ainda que os dados não são carregados para um meio físico, mas funcionam como que se estivessem num único BD.

A transferencia dos dados é feita apartir do XML (Hoffer *et al.*, 2010), onde são criados *schemas*, mas esses não são preenchidos com dados reais (Liu & Özsu, 2009).

A principal função da “ (...) federação de dados consiste basicamente em criar uma agregação de várias fontes de dados existentes, em uma única fonte de dados, de forma que os serviços e aplicações façam uso desta nova fonte de dados, que provê uma visão homogênea dos dados.” (Ferreira R. , 2007).

Segundo Josko (s.d.) e Hoffer *et al.* (2010), esse mecanismo é utilizado pelas tecnologias de Integração de Informações Empresariais (**EII** - *Enterprise Information Integration*). A figura a seguir mostra um exemplo da arquitectura da federação de dados e a suas tecnologias.

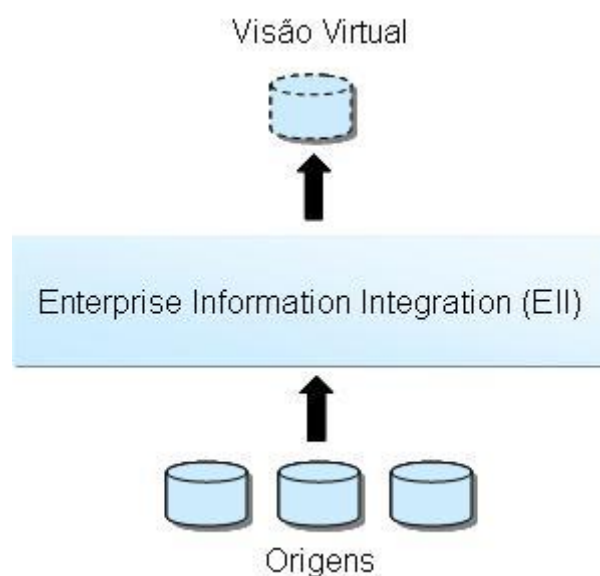


Figura 8 - Arquitetura de Virtualização dos Dados e as Tecnologias que a apoiam

Fonte: (Josko, s.d.)

Essa técnica é parecida com a de *Data Warehousing*, mas a diferença está na forma como é feita a movimentação dos dados, ou seja, a federação de dados, como referido anteriormente, ela não usa uma movimentação física dos dados entre as bases, em suma, “trata-se de uma base de dados lógica, que dinamicamente lê as demais fontes de dados e constitui uma visão homogénea e transformada dos dados.” (Ferreira R. , 2007).

Contudo, esta técnica tem alguns problemas, e segundo Ferreira (2007), esses são causados se alguma fonte ficar inoperante, pois existe um alto grau de conexão entre a base de dados virtual e as bases reais.

Essa técnica não deve ser utilizado em “ (...) situações cujo nível de qualidade dos dados seja problemático, pois há o potencial impacto sobre o tempo de resposta devido ao aumento da concorrência sobre as bases de origem.” (Josko, s.d.).

Políticas de SLA e monitoração destes recursos devem ser empregadas em conjunto com a técnica de estruturação da federação de dados, a fim de garantir a disponibilidade da fonte de dados virtual, além de que, realizar o estudo de escalabilidade e desempenho, pois esta nova base virtual será acessada concorrentemente, por vários serviços e processos, supostamente. (Ferreira R. , 2007).

6.2.3 Propagação de Dados

De acordo com Hoffer *et al* (2010), esta técnica faz a duplicação dos dados que se encontram nas fontes de dados. Ela faz a cópia dos dados de um local para outro quase em tempo real (Josko, s.d.), e isso é a sua principal vantagem, segundo Hoffer *et al.* (2010).

Sempre que houver um novo dado, esse será actualizado usando uma forma de propagação orientada a eventos (Hoffer *et al.*, 2010).

De acordo com Hoffer *et al.* (2010) e Josko (s.d.) as atualizações podem ser:

- **Síncronas** – todos os dados através da rede são constantemente actualizados de modo que um utilizador em qualquer local pode acessar dados em qualquer lugar

na rede, a qualquer momento e obter a mesma resposta. Enquanto todos as atualizações não forem copiadas, a transacção não será terminada.

- **Assíncronas** – separa as atualizações para as cópias remotas, ou seja, as cópias de dados replicados serão mantidas em diversos nós para que os servidores possam aceder a esses dados sem que seja através da rede.

Essa técnica utiliza tecnologias como **Integração de Aplicações Empresariais (EAI)**, **Replicação de Dados da Empresa (EDR)** e **ETL Tempo Real (R-ETL - Real-Time ETL)** (Hoffer *et al.*, 2010; Josko, s.d.).

“Enquanto as tecnologias de EDR e R-ETL – centradas em dado – oferecem maior capacidade no tratamento de grande volume de dados, a tecnologia de EAI, centrada na troca de mensagens, viabiliza o tráfego bidirecional de pequenas porções de dados entre dois pontos (...).” (Josko, s.d.).

A figura mostra um esboço da arquitetura de propagação de dados.

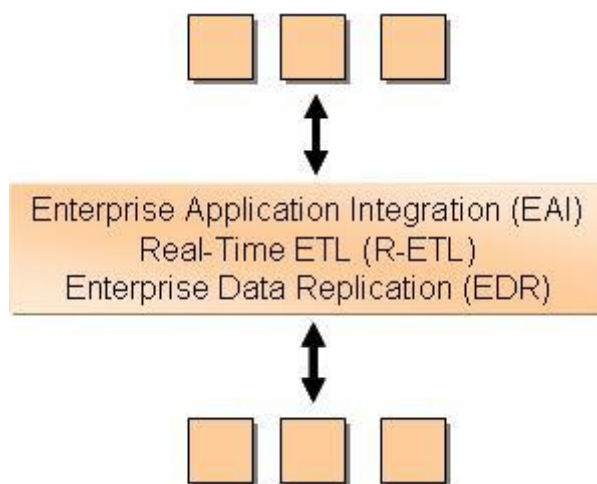


Figura 9 - Arquitetura de Propagação de Dados e as Tecnologias que a apoiam

Fonte: (Josko, s.d.)

6.2.4 Comparação das três abordagens

As principais diferenças entre essas arquiteturas se baseiam na forma como os dados são transmitidos e o tempo que leva para que os dados estejam atualizados.

A seguinte tabela faz uma comparação detalhada das três abordagens para ID.

Métodos	Vantagens	Desvantagens
Consolidação (ETL)	<ul style="list-style-type: none"> Utilizadores estão isolados das cargas de trabalho contraditórias sobre sistemas de origem, especialmente atualizações. É possível manter a história, não apenas os valores atuais. Um armazenamento de dados projetado para necessidades específicas podem ser acessados rapidamente. Funciona bem quando os escopos das necessidades de dados são previstos com antecedência. Transformações de dados podem ser agrupados para uma maior eficiência. 	<ul style="list-style-type: none"> Custos de manutenção da rede, do armazenamento, e dos dados podem ser elevados. O desempenho pode degradar-se quando o <i>Data Warehouse</i> torna muito grande (com determinadas tecnologias).
Federação (EII)	<ul style="list-style-type: none"> Os dados são sempre atual (como visualizações relacionais), quando solicitado. É simples para o aplicativo de chamada. Ele funciona bem para aplicativos somente de leitura porque apenas os dados solicitados precisam ser recuperados. 	<ul style="list-style-type: none"> Cargas de trabalho pesadas são possíveis para cada solicitação, devido à realização de todas as tarefas de integração para cada solicitação. Acesso de gravação para fontes de dados podem não ser suportado.

	<ul style="list-style-type: none"> • É ideal quando as cópias de dados de origem não são permitidas. • ETL dinâmico é possível quando não se pode antecipar as necessidades de integração de dados previamente ou quando há a necessidade de uma só vez. 	
Propagação (EAI & ERD)	<ul style="list-style-type: none"> • Os dados estão disponíveis em tempo quase real. • É possível trabalhar com ETL para <i>Data Warehousing</i> em tempo real. • Acesso transparente está disponível para fonte de dados. 	<ul style="list-style-type: none"> • Existe uma considerável sobrecarga (mas em segundo plano) associado com a sincronização de dados duplicados.

Tabela 3 - Comparação das três abordagens para *ID*Fonte: Adaptado (Hoffer *et al.*, 2010, p. 445)

7 Os Subsistemas de ETL

Todo sistema de ETL é constituído por uma arquitetura de 34 subsistemas críticos (Kimball *et al.*, 2008). Embora o artigo original foi publicado como “*The 38 Subsystems of ETL*”, esse ganhou uma nova versão atualizado por Bob Becker em Outubro de 2007, por isso passou a ser “*The Subsystems of ETL Revisited*” (Kimball & Ross, 2010, p. 430).

Como referenciado anteriormente, um sistema ETL consome quase 70% do tempo e recurso na implementação de um DW. É necessário sabermos o porquê que um processo ETL é tão complexo e consome todo esse tempo e recurso. Podemos constatar que muitas pessoas conhecem as letras: extrair os dados (E), transformar esses dados (T), e fazer o *load* desses dados transformados (L) (Kimball & Ross, 2010). Essa sessão mostra que um processo ETL, não fica “só” nas três fases anteriormente prescritas.

Becker (2007) afirma que é importante entendermos esses subsistemas para quando formos implementar o nosso sistema, ele seja eficaz, pois esses subsistemas são importantes para um projeto de DW.

Um sistema ETL é subdividido em quatro componentes fundamentais (Kimball *et al.*, 2008), onde cada componente é composto por vários subsistemas. Esses componentes são:

- **Extração** – recolha dos dados (Subsistema de 1 a 3);
- **Limpar e Ajustar os Dados** – os dados passam por um processo de transformação, para melhorar a sua qualidade (Subsistema 4 a 8);
- **Entrega dos dados** – os dados são carregados para o servidor para a apresentação (Subsistema 9 a 21);
- **Gerenciamento** – gestão dos sistemas e processos do ambiente ETL de uma forma coerente (Subsistema 22 a 34).

7.1 Extração: Coletar os Dados

A parte inicial de qualquer sistema de ETL é a coleta dos dados, onde esses são extraídos de uma ou várias fontes (origens), e podem ou não passar por um processo de transformação. Essa etapa de extração de dados é constituída por três subsistemas, e elas são (Kimball & Ross, 2010; Lima, 2010; Becker, 2007):

- **Perfil dos dados** (Subsistema 1) – é uma análise técnica dos dados para descrever seu conteúdo, consistência e estrutura. Ela desempenha duas funções: a estratégica e a tática. A estratégica, depois de identificados as fontes de dados, estes devem ser avaliados/analísados a fim de saber se serão uteis ou não, ou seja, se vão suportar a missão pretendida, pois, se tal não acontecer, uma equipa pode vir a deparar que essa fonte não era útil, e como cada segundo é muito num projeto, o importante é fazer o correto. A tática tenta identificar todos possíveis problemas que podem ocorrer.

Essa análise ajuda a equipa ETL com a limpeza de dados, pois como os dados “sujos” não serão extraídos, a equipa ganha tempo para preocupar com assuntos mais importantes do projeto (Kozielski & Wrembel, 2008).

- **Captura de Dados Alterados (CDC)** (Subsistema 2) – isola as mudanças ocorridas nas fontes de dados, ou seja, transfere somente as mudanças relevantes ocorridas desde a última atualização. Ela é utilizada no [*load incremental*](#)⁵. Também utiliza *logs* e compara os registos baseados nos algoritmos de CRC.
- **Sistema de Extração** (Subsistema 3) - Extração e movimentação dos dados de origem para dentro do DW ou outro sistema de destino, para processamento futuro.

7.2 Limpar e Ajustar os Dados

Nem sempre os dados extraídos estão limpos, pois são dados brutos (dados sujos) que precisam ser transformados conforme as necessidades do sistema que se pretende implementar. Esses dados passam por uma limpeza e ajustes, ou uma padronização para que possam ser dados de qualidade.

Essa etapa é constituída por cinco subsistemas, são eles (Kimball & Ross, 2010; Lima, 2010; Becker, 2007):

- **Sistema de limpeza de dados** (Subsistema 4) – Implementa processos de qualidade de dados para identificar violações de qualidade.
- **Acompanhamento de erro** (Subsistema 5) – Captura todos os ‘eventos de erro’, que serão as entradas vitais para a melhoria da qualidade dos dados.
- **Criação de Dimensão de auditoria** (Subsistema 6) – Junta Metadados para cada Tabela Fato, como uma dimensão. Esses metadados estão disponíveis para aplicações de BI para a visibilidade na qualidade de dados.

⁵ Ver load incremental

- **Tirar a duplicidade de dados** (Subsistema 7) – Elimina dados redundantes de dimensões, como clientes ou produtos. Pode requerer integração cruzada entre múltiplas origens e a aplicação de regras para identificar qual a versão mais correta de uma linha duplicada. Muitas vezes utiliza a lógica *Fuzzy*.
- **Conformidade de dados** (Subsistema 8) - Conformidade consiste de todos os passos necessários para alinhar o conteúdo de algumas ou de todas as colunas em uma dimensão com colunas em dimensões semelhantes ou idênticas em outras partes do DW. Ela tenta garantir que os conteúdos referentes ao mesmo assunto estejam nos conformes, por isso ela é responsável por criar e manter dimensões e fatos conformados.

7.3 Entrega: Preparação para Apresentação

Nessa fase os dados são preparados para serem carregados para o servidor. Segundo Kimball *et al* (2008), a missão inicial do sistema de ETL é a entrega das tabelas de dimensão e de fatos.

O processo de entrega é constituído por técnicas bem definidas e disciplinadas, que precisam ser bem utilizados na implementação do DW dimensional de sucesso que seja confiável, escalável e sustentável (Kimball *et al*, 2008).

Os subsistemas que constituem a entrega são (Kimball & Ross, 2010; Lima, 2010; Becker, 2007):

- ***Slowly Changing Dimension (SCD) Manager*** (Subsistema 9) – quando houver alteração nas dimensões, o sistema tem de ser capaz de lidar com essa alteração, usando a lógica de SCD. Essa é um dos elementos mais importantes numa arquitetura ETL.
- **Criar as chaves substitutas** (Subsistema 10) – Mecanismo robusto para a produção de chaves substitutas, de forma independente para cada dimensão. Independente da instância de base de dados, capaz de atender os clientes distribuídos.

- **Gestor de Hierarquia** (Subsistema 11) – verificação da validade dos dados e manutenção do sistema para todas as formas de hierarquia desiguais de profundidade indeterminada.
- **Gestor de Dimensões Especiais** (Subsistema 12) – Cria locais - espaços reservados na estrutura de ETL para sustentar os processos repetitivos específicos da organização, no desenho de dimensões específicas como as *Junk Dimensions*, *Mini Dimensions* e indicadores de comportamento.
- **Construtores de Tabela de Fatos** (Subsistema 13) – Construção dos três tipos básicos de tabela fato: Transacional, Periódico e Cumulativo (*transaction grain*, *periodic snapshot* e *accumulating snapshot*).
- **Pipeline Chave Substituta** (Subsistema 14) – substituir a chave natural operacional das tabelas de origem pelas chaves substitutas. Essas chaves substitutas pertencem as dimensões e serão utilizadas para o relacionamento com as tabelas fato. Isso significa que para cada chave estrangeira na tabela de fato, existe uma entrada na tabela de dimensão correspondente.
- **Construtor de Tabela Ponte Multi-valorizado** (Subsistema 15) – Construção e Manutenção das tabelas ponte para suportar os relacionamentos multi-valorizados.
- **Manipulador de dados tardios** (Subsistema 16) – Aplica-se a modificações especiais para os procedimentos de processamento padrão para lidar com dados de fato e de dimensão tardios.
- **Gestor de dimensão** (Subsistema 17) – é uma autoridade centralizada que prepara e publica dimensões adequadas para a comunidade de DW.
- **Provedor de tabela de fatos** (Subsistema 18) – esse provedor utiliza as dimensões adequadas fornecidas pelo gestor de dimensão. É responsável pela

criação, manutenção, e uso das tabelas de fatos, ou seja, ele é que administra essas tabelas.

- **Construtor de agregações** (Subsistema 19) – constrói e mantém as agregações para melhorar a performance nas consultas. As agregações são iguais a Indexes⁶.
- **Construtor de cubo OLAP** (Subsistema 20) – seleciona os dados do *schema* dimensional para popular os cubos OLAP.
- **Gestor de propagação de dados** (Subsistema 21) - prepara dados conformados e integrados no servidor de apresentação do *Data Warehouse*, para entrega em outros ambientes, para propósitos especiais.

7.4 Gerenciamento do Ambiente de ETL

Um DW de sucesso é uma fonte confiável para as tomadas de decisões nas empresas (Kimball & Ross, 2010). Mas para que seja de sucesso, o sistema de ETL deve alcançar três objectivos como a confiabilidade, a disponibilidade e a capacidade de gerenciamento (Kimball & Ross, 2010, p. 433; Kimball *et al*, 2008).

A confiabilidade nesse caso, o processo ETL tem que fornecer dados em tempo oportuno que seja confiável em qualquer nível de detalhe (Kimball *et al.*, 2008).

Na mesma linha Kimball *et al* (2008) diz que a disponibilidade quer dizer que o DW tem que estar pronto e disponível como prometido.

É preciso uma capacidade de gerenciamento, pois um *Data Warehouse* de sucesso nunca termina, ele cresce constantemente e muda juntamente com o negócio, por isso os processos de ETL devem evoluir normalmente também (Kimball *et al.*, 2008).

⁶ *Index* - são estruturas de dados específicos criados para melhorar o desempenho (Kimball *et al*, 2008).

Os seguintes subsistemas de gerenciamento ajudam a atingir esses objectivos (Kimball & Ross, 2010; Lima, 2010; Becker, 2007):

- **Agendamento de tarefas** (Subsistema 22) – A estratégia de gerenciamento da execução dos ETL deve ter um mecanismo confiável, incluindo os relacionamentos e dependências entre as tarefas de ETL.
- **Sistema de Backup** (Subsistema 23) – assim como qualquer sistema, DW está sujeito a riscos, por isso é importante manter uma cópia do ambiente de ETL para uma possível recuperação, *restart* e arquivamento.
- **Recuperação e Restart** (Subsistema 24) – um sistema em produção está sujeito a falhas, por isso é necessário sempre fazer *backups*. Quando houver uma falha de sistema, temos que ter processos capazes de fazer uma recuperação (a partir do ultimo *backup*), ou reiniciá-la.
- **Controlo de Versão** (Subsistema 25) – tira “*snapshots*” das versões de ETL e a mantém arquivadas, para eventual recuperação das lógicas e metadados do “*ETL pipeline*”.
- **Migração de Versão** (Subsistema 26) – migração de uma versão completa do “*ETL pipeline*” a partir do ambiente de desenvolvimento para um ambiente de testes e, finalmente, para o ambiente de produção.
- **Monitoramento do fluxo de trabalho** (Subsistema 27) – serve para monitorar a execução das tarefas, ou seja, se os *load* estão a ser conforme agendados. Painel de controlo e sistema de relatório de todas as tarefas agendadas (e.g. números de registos processados, resumos de erros e ações tomadas).
- **Ordenar** (Subsistema 28) – é uma das funções/capacidades fundamentais do processo ETL, por isso merece uma melhor atenção, pois garante a alta performance.

- **Linhagem e Dependência** (Subsistema 29) – identificam a origem de um elemento de dado e todos os pontos intermédios e transformações para esse elemento ou, por outro lado, começar com um elemento de dado específico em uma tabela de origem e revela todas as atividades realizadas nesse elemento de dado.
- **Problema de Escalonamento** (Subsistema 30) – estrutura de suporte que encaminha os problemas de ETL para o nível de solução apropriado.
- **Parallelizing e Pipelining** (Subsistema 31) – ajudam o processo ETL a terminar o *load* dentro do prazo estipulado. As vezes, esse é o maior desafio de um processo ETL (nas grandes empresas com grande volume de dados para serem entregues). Permite ao sistema de ETL a potencializar automaticamente o uso de múltiplos processadores ou computação em grade (*grid computing*) para entregas dentro dos prazos restritos.
- **Segurança** (Subsistema 32) – garantir acessos autorizados aos dados de ETL e metadados, de uma forma individual ou grupos (funções), e manter sempre o histórico dos acessos.
- **Gestor de Conformidade** (Subsistema 33) – Suporta os requerimentos organizacionais de conformidade, através, tipicamente, da manutenção da custódia da cadeia de dados e do acompanhamento dos acessos aos dados (quem teve o acesso autorizado ao dado).
- **Repositório dos Metadados** (Subsistema 34) – Captura os metadados do ETL, incluindo os metadados de processo, metadados técnicos e metadados do negócio que significam todos os metadados do ambiente de DW/BI.

8 Os Metadados

O dado é atualmente um recurso muito valioso numa organização, pelo que é importante que ele esteja bem organizado. A tecnologia de metadados surgiu para dar

suporte as organizações no que toca a um melhor conhecimento dos dados que possuem (Vaz, sd).

Os metadados são muito importantes e cada vez estão a ser desenvolvidas ferramentas que tenham as soluções de metadados embutidas, e também muitos livros e artigos estão sendo escritos sobre as melhores práticas em relação à estratégia de metadados (Kimball & Caserta, 2004).

Nesta seção vamos falar dos metadados referentes ao ETL porque existe diversos tipos de metadados.

8.1 O que são Metadados

Segundo Kimball & Caserta (2004), a exata definição dos metadados é ambíguo, e é uma tarefa difícil definir exatamente o que ele é. Por isso podemos encontrar várias definições sobre metadados.

Para Hoffer *et al.* (2010), são dados que descrevem as propriedades ou características dos dados do utilizador e do contexto desses dados. Podemos encontrar autores que preferem dizer que metadados “são dados sobre dados”, “dados que descrevem dados”, ou “informações dos conteúdos dos dados” ou dados de alto nível que descrevem dados de nível inferior (Souza, 2003).

São informações estruturadas que descreve, explica, localiza, ou de outra forma torna mais fácil para recuperar, usar ou gerenciar um recurso de informação (Niso, 2004).

Um metadado de um item de dados, mostra como esse item de dado foi criado, em que contexto pode ser utilizado, como foi transformado, ou como pode ser interpretado ou processado (Wang, 2008).

Segundo Mussi (2004), sem os metadados, os dados não têm significados. Logo é importante termos uma documentação dos dados, pois como os dados crescem

exponencialmente numa organização, isso ajudará a conter a consistência dos dados e evitar os possíveis problemas que podem surgir.

Por isso devemos utilizar os metadados, porque, eles fornecem recursos necessários para entender os dados através do tempo.

Kimball & Ross (2010) disseram que essa ideia de metadados começou a esclarecer e daí começou-se a falar dos **metadados “back room”** que guia o processo de Extração, Transformação (limpeza), e *Load*, como também dos **metadados “front room”** que faz com que as nossas ferramentas de consultas e relatórios funcionar sem problemas.

Metadados “back room” são metadados que ajudam o DBA a trazer os dados para o DW, e também pode ser útil quando os utilizadores de negócios querem saber de onde veio os dados, ou seja, a origem dos dados (Kimball & Ross, 2010).

Metadados “front room” são metadados que beneficiam os utilizadores de negócios, e além de permitir que as ferramentas funcionem sem problemas, esse tipo de metadados é como um dicionário que contém todo o conteúdo do negócio representados por todos os elementos de dados (Kimball & Ross, 2010).

8.2 Metadados ETL

Kimball & Caserta (2004) disseram que o maior desafio da equipa de ETL em relação aos metadados está em como e onde armazenar o processo-fluxo da informação. Mas as ferramentas de ETL conseguem manter esse fluxo de metadados automaticamente. O problema seria maior em caso de codificação manual, pois teríamos que implementar nosso próprio repositório central de metadados.

Como tínhamos referido anteriormente vamos falar somente dos metadados em relação ao ETL, por isso vamos listar alguns metadados que compõem o ETL. Segundo Kimball & Caserta (2004), esses metadados⁷ estão subdivididos em:

- **Metadados de Fontes de Origem;**
- **Metadados *Data-Staging*;**
- **Metadados DBMS; e**
- **Metadados *Front Room*.**

⁷ Para mais informações sobre esses metadados consulta o livro (Kimball & Caserta, The data warehouse ETL toolkit : practical techniques for extracting, cleaning, conforming, and, 2004)

Capítulo 2: As Ferramentas de ETL

1 Introdução

Neste capítulo vamos falar sobre as ferramentas de ETL, desde os primórdios até os dias de hoje. Também vamos falar das ferramentas *OpenSource* e as comercializadas, bem como o que faz uma ferramenta de ETL, suas características, e benefícios. No final vamos falar sobre a ferramenta TOSDI.

Neste capítulo não vamos descrever as ferramentas de consultas e análise pois, o foco desse trabalho é a integração dos dados e não como esses vão ser consultados.

Este capítulo é muito importante para uma maior compreensão do caso de estudo que vamos apresentar no próximo [Capítulo 3](#).

2 Conceito

De acordo com Kimball *et al* (2008), é uma aplicação de *software* utilizado para desenvolver e entregar sistemas de ETL. Ferramentas de ETL são aplicações de *software* cuja função é extrair dados transacionais de diversas fontes ou origens, transformar esses dados para garantir padronização e consistência as informações e carregá-los para um ambiente de consulta e análise, conhecido como *Data Warehouse* (Neto, 2012).

Na mesma perspectiva K (2012) disse que essas ferramentas são destinadas a extração, transformação, e *load* dos dados para o DW para a tomada de decisão, mas é importante aqui não pensarmos que essas ferramentas só servem para construção do DW.

Com uma grande quantidade de transformação e integração que essas ferramentas possuem, pode-se fazer sistemas com as mais variadas funcionalidades.

Inicialmente todo trabalho era feito manualmente, o que era cansativo e complexo e muitas das vezes o resultado esperado nem era satisfatório, por isso, surgiram as ferramentas de ETL para eliminar esses problemas (K, What does ETL tool mean?, 2012).

Contudo podemos constatar que as funcionalidades de uma ferramenta ETL não se limitam ao subprocesso de um DW, pois podemos encontrar ferramentas que possuem outras funcionalidades como:

- Integração dos dados;
- Carregamento de dados de um BD para um ficheiro/arquivo em vários formatos;
- Ler vários formatos de arquivos e inserir num BD.

De acordo com Neto (2012), essas ferramentas de ETL ainda disponibilizam recursos, como: geração de metadados, conectividade com os principais SGBD, leitura de arquivos como XML, Folhas de Cálculos, e ficheiros texto (TXT), possuem também funções que auxiliam na transformação de dados, reutilização de códigos, gerenciamento centralizado de projetos, possui diagramas que ajudam no desenvolvimento, documentação facilitada, entre outros.

Ainda segundo Neto (2012, p.46), essas ferramentas possuem como principais características:

(...) Conectividade com as principais Base de Dados, com arquivos simples e planilhas; suportar as principais plataformas de *hardware* e sistemas operacionais; possuir recurso de depuração como *breakpoint* e execução passo-a-passo; componentes para manipulação de *String* (agregar, desagregar, limpar), funções matemáticas, execução de scripts com inserção de códigos pelo desenvolvedor sendo SQL, Java, Perl ou, até mesmo, linguagem proprietária como é o caso do CloverETL; administração e gerenciamento do projeto; e uma boa interface gráfica e intuitiva.

Atualmente as empresas estão a apostar no uso dessas ferramentas para poder dar resposta ao mercado e ficar cada vez mais competitivo. Por isso podemos encontrar vários fornecedores dessas ferramentas.

De entre as ferramentas existentes no mercado moderno, temos as comerciais e as *Opensource*.

Segundo Thoo *et al* (2012), numa pesquisa da Gartner sobre *Ferramentas de Integração de Dados*, de entre as “melhores” comerciais podemos encontrar a Informática e a IBM, e entre os *OpenSource* podemos encontrar o Talend e o Pentaho. Sendo que não podemos afirmar que uma seja melhor que a outra, porque a ferramenta adequada depende da necessidade cliente.

3 A Evolução das Ferramentas de ETL

Inicialmente tudo era armazenado no *mainframe*, mas com o começo e evolução dos sistemas de computadores de mainframes monolíticos para sistemas de computação distribuídos, deu-se início a primeira tentativa de *Business intelligence*, e houve a introdução das primeiras soluções de ETL (Montcheil & Dupupet, 2006).

A evolução do *Business Intelligence* proporcionou um grande avanço das ferramentas de ETL (Kakish & Kraft, 2012). Por isso Ferreira *et al*, (2010) disseram que temos 3 gerações dessas ferramentas (Ver Tabela 1).

Como tínhamos referenciado no capítulo anterior, as primeiras formas ETL, apareceram nos meados dos anos 90 e eram uma forma de codificação manual, de onde deu início a primeira geração de ETL (Ver Tabela 1).

Essas primeiras ferramentas, extraíam os dados do *mainframe* e fazia *load* desses dados numa BD de destino (Zode, 2011). Essa codificação foi uma grande solução ETL na altura, já que os dados se encontravam num computador de grande porte (*Mainframe*), por isso, seria mais fácil introduzir código nela e fazer a consulta dos dados que ali se encontravam (Zode, 2011).

Para resolução desse problema de codificação manual, criaram soluções de ETL, ou seja, ferramentas para integração de dados.

No capítulo anterior falamos um pouco sobre história do ETL, mas nessa sessão vamos falar sobre as diferentes gerações que o ETL passou até os dias de hoje.

Segundo Kakish & Kraft (2012), Montcheil & Dupupet (2006) e Zode (2011) temos as seguintes gerações de ETL:

- **1ª Geração – Origem de ETL e geradores de códigos;**
- **2ª Geração – Mecanismos de ETL**
- **3ª Geração – Arquitetura de ETL**

3.1 Primeira Geração de ETL: Geradores de Códigos

Nos meados da década de 1990, os vendedores começaram a desenvolver ferramentas de ETL para resolver o problema de escrever os códigos manuscritos complexos, e esses produtos que começaram a desenvolver eram geradores de códigos legados (Zode, 2011).

O original da ferramenta de integração de dados gerava códigos nativos (Montcheil & Dupupet, 2006), ou seja, essas ferramentas ETL foram escritas em códigos nativos da

plataforma de sistema operacional e que só executa no sistema operacional nativo (Kakish & Kraft, 2012).

Segundo os autores Kakish & Kraft (2012) e Montcheil & Dupupet (2006), a maioria dessas ferramentas geradoras de código gerava COBOL, isso porque, os dados eram armazenados em grande parte no *mainframe*.

A Figura a seguir mostra um exemplo das ferramentas baseadas em códigos.



Figura 10 - Ferramentas baseadas em códigos

Fonte: Adaptado (Zode, 2011, p. 6)

Programa de extração foi gerado automaticamente como código-fonte que foi compilado, agendado e executado em modo *batch* (Zode, 2011). Estas ferramentas usavam uma *thread* único e não suportava o paralelismo (Zode, 2011), entretanto o desempenho era bom.

Na época essa alternativa funcionou na integração de dados e o desempenho era muito bom, mas o problema era a manutenção (Kakish & Kraft, 2012). E do mesmo modo pensa Montcheil & Dupupet (2006), que esses produtos ajudaram na integração de dados, pois passou a ser mais fácil do que era antes, visto que, em vez de escrever manualmente o programa para essa integração de dados, essa ferramenta centralizada

gerava o processo de integração de dados e ainda distribuía o código para outras plataformas.

De acordo com Montcheil & Dupupet (2006), o desempenho era ótimo, mas só se os dados fossem armazenados em arquivos simples (*flat files*), e BD hierarquizados, e o acesso de nível de registo for rápido.

Os autores ainda afirmam que essas ferramentas exigiam um profundo conhecimento de programação em diferentes plataformas. Um outro problema, como tínhamos falado anteriormente, seria a manutenção, já que os códigos estavam espalhados entre diferentes plataformas e diferem com o tipo de fontes e destinos (Montcheil & Dupupet, 2006).

Do mesmo modo Zode (2011) disse que esses geradores de códigos exigiam uma intensa programação em COBOL ou C.

Como nessa época ainda não tinha aparecido os computadores, e o *mainframe* era bastante utilizado para armazenamento de dados, essa ferramenta funcionou bem, mas usando uma abordagem de BD relacional, isso não seria bem-sucedida, visto que não consegue gerir grandes volumes de dados (Montcheil & Dupupet, 2006).

Vamos citar alguns pontos fortes e fracos que marcaram essa geração, segundo Zode (2011):

➤ **Pontos Fortes**

- Capaz de extrair dados de sistemas legados.
- O desempenho foi bom porque herdaram o desempenho de compilação de código nativo.

➤ **Pontos Fracos**

- Requer um conhecimento profundo de programação em diferentes plataformas.
- Demonstra menos sucesso em BD relacional para lidar com grande volume de dados.
- Não suporta processamento paralelo.
- Muitas transformações requerem codificação manual.

Temos então com exemplo das ferramentas da primeira geração (Zode, 2011):

- **SAS/Warehouse Administrator** desenvolvido por **SAS Institute Inc.** em **1997**.
- **Prism** por Prism Solutions.
- **Passport** por Apertus Carleton Corp
- **ETI-EXTRACT Tool Suite** por Evolutionary Technologies
- **Copy Manager** por Information Builders

3.2 Segunda Geração de ETL: Mecanismos de ETL

Foi nos meados do fim de 1990's que os fornecedores começaram a fornecer os primeiros mecanismos básicos de ETL para automatizar o processo ETL (Zode, 2011), depois da experiência com os geradores de códigos.

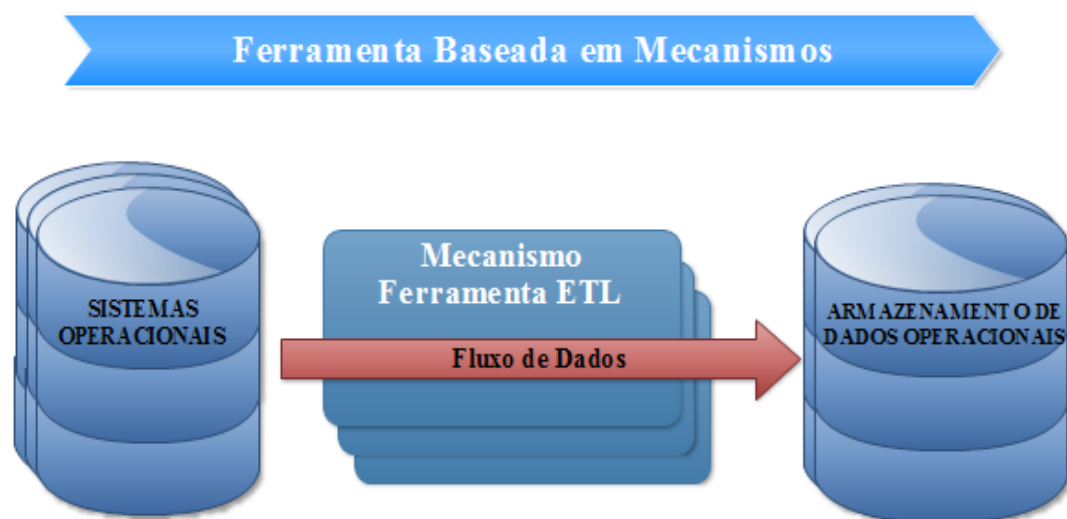


Figura 11 - Ferramentas baseadas em Mecanismos

Fonte: Adaptado (Zode, 2011, p. 7)

Segundo Kakish & Kraft (2012) e Montcheil & Dupupet (2006), esse mecanismo de ETL executa todo o processo de transformação.

Diferente da primeira geração, nessa geração o processo ocorre no mecanismo e não no sistema de origem (Zode, 2011).

E o mecanismo básico da ferramenta de ETL gere os processos ETL através de um mecanismo interno, por isso ele não gera códigos como os da primeira geração (Zode, 2011).

Se na primeira geração os desenvolvedores tinham que programar muitos códigos, na segunda geração esses desenvolvedores terão menos trabalho, pois em vez de saberem muitas linguagens de programação, precisam somente de ter o conhecimento de uma linguagem de programação: **a programação ETL** (Kakish & Kraft, 2012).

Nessa geração as ferramentas possuíam um ambiente gráfico onde os desenvolvedores desenhavam os *workflows*, e era esses *workflows* que diziam como os dados iam ser processados (Zode, 2011).

De acordo com Kakish & Kraft (2012), os dados vêm de diferentes fontes de heterogêneos e passam pelo mecanismo de ETL linha a linha e são armazenados num sistema de destino.

Como esse processamento é feito linha por linha, faz com que fosse mais lenta, e a geração de programas de ETL sofria de uma sobrecarga de alto desempenho (Kakish & Kraft, 2012).

Alguns desses mecanismos suportam o processamento paralelo mas, para que isso aconteça é necessário a criação de uma partição no gestor de servidor (Zode, 2011).

Zode (2011) disse ainda que nas ferramentas baseadas em mecanismos, o repositório de metadados é mantido em diferentes servidores que precisam ser conectados a partir da aplicação cliente.

Vamos citar alguns pontos fortes e fracos que marcaram essa geração, segundo Zode (2011):

➤ **Pontos Fortes**

- Possui um ambiente gráfico e com recursos de transformação.
- Essa abordagem baseada em mecanismo é rápido, eficiente e *multi-thread*.
- As funções de ETL estão altamente integrados e automatizados.
- As funções suportadas incluem agenda, monitoramento, agregação e transformação.

➤ **Pontos Fracos**

- Todos dados que vem de várias fontes, tem que passar pelo mecanismo que processa a transformação desses dados linha por

linha, o que faz com que seja muito lento com um grande volume de dados.

- O mecanismo que executa toda a transformação tornou-se um problema no processo de transformação.
- Ferramentas baseadas em mecanismos requerem administradores dedicados que configurem a plataforma de mecanismo para fornecer um alto desempenho.

Nessa geração apareceram algumas ferramentas⁸ como:

- **Power mart 4.5** desenvolvida por **Informatica Corporation** em 1999 (Zode, 2011).
- **Ardent DataStage** desenvolvida por **Ardent Software, Inc.** agora da **IBM corp.**
- **Data Mart Solution** por **Sagent Technology**
- **Tapestry** por **D2K**

3.3 Terceira Geração de ETL: A Arquitetura de ETL

Aproveitando os pontos fortes das duas gerações anteriores e tentar dar resposta aos desafios pelo que passaram essas gerações, assim surgiram as ferramentas de ETL da terceira geração, a mais recente: a arquitetura de ETL (Extração, Transformação e *Load*) (Montcheil & Dupupet, 2006).

⁸ Disponível em: <http://pt.scribd.com/doc/65518203/31/First-Generation-ETL-Tools-%E2%80%93-Examples>, consultado a 09 de Junho de 2013

No início da segunda geração os fornecedores de BD começaram a investir grandes quantidades de recursos para melhorar significativamente as capacidades de suas linguagens SQL (Montcheil & Dupupet, 2006).

Com as melhorias conseguidas, eles tornaram possíveis para uma ferramenta ETL gerar e executar os processos de integração de dados altamente otimizados, no SQL nativo ou nas outras linguagens das BD envolvidas nestes processos (Montcheil & Dupupet, 2006).

Zode (2011) disse que houve muitas melhorias desde 1990's com os geradores de códigos, passando pelas ferramentas baseadas em mecanismos, até a terceira geração das ferramentas ETL.

Essas ferramentas, de acordo com Montcheil & Dupupet (2006), tem um ambiente altamente gráfica, e com capacidade para gerar SQL nativo para executar as transformações de dados no servidor de DW. Na perspectiva de Zode (2011), essas ferramentas fornecem uma interface gráfica de fácil utilização que permite o desenvolvedor trabalhar sem mesmo precisar de um treinamento.

Na mesma linha Kakish & Kraft (2012) afirmam que essa arquitetura possui capacidade para gerar SQL nativo, e elimina o servidor *hub* entre o sistema de origem e destino.

Antigamente apenas COBOL foi utilizado, nos quais ferramentas geradores de códigos eram código fonte, mas atualmente, os fornecedores estão apoiando várias plataformas em que as ferramentas baseadas em códigos podem gerar seu próprio código (Zode, 2011).

A maioria das ferramentas geradores de códigos estão utilizando SQL para gerar os códigos e também suportando todos recursos ETL (Zode, 2011). A parte mais positiva do gerador de código é compilação do código produzido que pode executar em qualquer plataformas, esse código compilado não só é rápido como também distribui o *load* entre múltiplas plataformas para aumentar o desempenho (Zode, 2011).

Mas também as ferramentas ETL baseadas em mecanismos estão surgindo muito rápido, e com capacidade de executar transformação complexa a uma velocidade mais rápida de execução (Zode, 2011).

Segundo Zode (2011), as ferramentas de ETL disponíveis nessa geração são ricas em recursos de transformações e normalmente suporta muitos inputs e/ou outputs de BD, ficheiros simples (*flat files*), geração da chave substituta, várias funções de transformação, etc.

Ainda, Kakish & Kraft (2012) acrescentam que reduz o tráfego da rede para aumentar o desempenho, distribui o *load* entre Base de Dados para melhorar a escalabilidade, suporta todo tipo de fonte de dados. Também elimina a sobrecarga de desenvolvimento e manutenção das rotinas complexas e transformação em *workflow* ETL (Zode, 2011).

A fase de transformação usando relacional DBMS, faz processamento de dados melhor do que na segunda geração que era feita linha por linha. E como RDBMS suporta a integração de dados, essas ferramentas aproveitam dessa capacidade para melhorar o seu desempenho (Kakish & Kraft, 2012).

As ferramentas possuem um repositório de metadados interno que pode ser diferente do repositório de metadados do DW (Zode, 2011). Também possui recursos como, monitoramento, agendas, carregamento em massa, agregação incremental, etc. (Zode, 2011).

Vamos citar alguns pontos fortes e fracos que marcaram essa geração, segundo Zode (2011):

➤ **Pontos Fortes**

- As ferramentas de ETL disponíveis atualmente podem lidar com transformações mais complexas a velocidade mais rápida de execução.

- O código gerado pela ferramenta baseada em código pode executar em várias plataformas em uma alta velocidade e faz com que as organizações distribuam *loads* entre múltiplas plataformas para aumentar o desempenho.
- Suporta muitos tipos de processamento paralelo.
- Capacidade de ler ficheiros XML muito eficiente.
- Fornece recursos como agregação incremental, *Slowly Changing Dimension* (SCD), normalização de fontes de dados. Basicamente suporta todo o conceito de DW como normalização, denormalização, SCD, etc.
- Possui recursos como controlo de versão que permite o desenvolvedor manter as versões do código fonte sem sobrescrever o código original, manutenção de metadados e documentação.
- Suporta também o *Debugger* que permite o desenvolvedor testar o código para rastreamento de defeitos.

➤ **Pontos Fracos**

- Não possui capacidade suficiente para suportar aplicações em tempo real.
- Algumas ferramentas estão limitadas para BD de origem como o OWB que suporta somente Oracle e ficheiros simples como fonte.
- Muitas das ferramentas de ETL recentes não suportam a integração a nível de metadados com ferramentas de utilizador final.

4 As Ferramentas de ETL Disponíveis no Mercado

Como visto na seção anterior as primeiras soluções ETL apareceram nos anos 90's. Daí muitos foram os fornecedores que quiseram criar ferramentas que suportassem o ETL, pois na época essas ferramentas começaram a ser lucrativas.

Desde o primeiro lançamento da ferramenta ETL até os dias de hoje muitos foram as ferramentas⁹ que apareceram e em várias versões, mas também muitos que apareceram recentemente podem não estar no mercado e se estiver é muito pouco utilizado.

Atualmente têm ferramentas capazes de processar milhões de dados em poucos segundos, e com diversos recursos de transformação dados, também podemos encontrar a maioria com um ambiente gráfico de desenvolvimento muito avançado, facilitando assim o trabalho do desenvolvedor.

Com uma ferramenta de ETL é possível fazermos um trabalho de um mês em um dia (ou uma hora para quem é mais otimista).

Atualmente pode-se encontrar uma grande variedade de ferramentas ETL, entre elas algumas são mais conhecidas (mais populares) e outras nem tanto. Também temos as que são comerciais e as que são *OpenSource*.

A tabela a seguir fornece-nos uma lista das ferramentas de ETL para integração de dados existentes.

Nº.	Ferramenta ETL	Página
1.	IBM Information Server	www.ibm.com
2.	PowerCenter Informatica	www.informatica.com
3.	Information Builders-iWay Software	www.informationbuilders.com

⁹ Podemos encontrar uma lista das ferramentas ETL que já existiram (ou ainda existem) no mercado aqui: <http://www.dbnet.ece.ntua.gr/~asimi/ETLTools.htm>, consultado a 08 de Junho de 2013

4.	SQL Server Integration Services (SSIS)	www.microsoft.com
5.	Oracle Warehouse Builder (OWB)	www.oracle.com
6.	Oracle Data Integrator (ODI)	www.oracle.com
7.	Pervasive Data Integrator	http://integration.pervasive.com
8.	Data Services	www.sap.com
9.	SAS Data Integration Studio	www.sas.com
10.	DMExpress	www.syncsort.com
11.	Talend Studio for Data Integration	www.talend.com
12.	Ab Initio	www.abinitio.com
13.	Adeptia	www.adeptia.com
14.	Alebra Technologies	www.alebra.com
15.	Altibase	www.altibase.com
16.	Apatar	www.apatar.com
17.	Arbutus Software	www.arbutussoftware.com
18.	Astera Software	www.astera.com
19.	Attunity	www.attunity.com
20.	Axway	www.axway.com
21.	BackOffice Associates	www.boaweb.com
22.	BIReady	http://biready.com
23.	C3 Business Solutions	http://c3businesssolutions.com
24.	CA Technologies	www.ca.com
25.	CDB Software	www.cdbsoftware.com
26.	Composite Software	www.compositesw.com
27.	DataRocket	www.datarocket.com
28.	DataStream	www.datastreams.co.kr
29.	Datawatch	www.datawatch.com

30.	DBSync	www.mydbsync.com
31.	Dell Boomi	www.boomi.com
32.	Denodo Technologies	www.denodo.com
33.	DFI	www.datafusion.ie
34.	Diyotta	www.diyotta.com
35.	ETI	www.versata.com
36.	ETL Solutions	www.etlsolutions.com
37.	Gamma Soft	www.gamma-soft.com
38.	GSS Group	www.gssgrp.com
39.	GT Software	www.gtsoftware.com
40.	HiT Software	www.hitsw.com
41.	HVR Software	www.hvr-software.com
42.	Innovative Routines International	www.iri.com
43.	Javlin	www.javlin.eu/en
44.	Jitterbit	www.jitterbit.com
45.	JumpMind	www.jumpmind.com
46.	Kapow Software	www.kapowsoftware.com
47.	Kinetic Networks	www.kineticnetworks.com e www.ketl.org
48.	Metatomix	www.versata.com
49.	Nimaya	www.nimaya.com
50.	Pentaho	www.pentaho.com e http://kettle.pentaho.com/
51.	Pitney Bowes Software	www.pb.com
52.	Progress Software	www.progress.com
53.	QlikTech	www.qlikview.com
54.	Quest Software	www.quest.com
55.	Red Hat	www.redhat.com

56.	RedPoint	www.redpoint.net
57.	Relational Solutions	www.relationalsolutions.com
58.	Safe Software	www.safe.com
59.	SchemaLogic	www.schemalogic.com
60.	Scribe	www.scribesoft.com
61.	Sesame Software	www.sesamesoftware.com
62.	SnapLogic	www.snaplogic.com
63.	Software AG	www.softwareag.com
64.	SQData	www.sqdata.com
65.	Stone Bond	www.stonebond.com
66.	Sypherlink	www.sypherlink.com
67.	Vision Solutions	www.visionsolutions.com
68.	WhereScape	www.wherescape.com

Tabela 4 - Lista das Ferramentas de Integração de Dados

Fonte: Adaptado (Thoo, Friedman, & Beyer, 2012)

Podem ainda existir outras ferramentas de ETL que não estão na tabela anterior, pois nós só estamos a falar das ferramentas de Integração de Dados. Só para constar existem ferramentas de ETL para Qualidade de Dados, Análise de Dados, etc.

4.1 As mais conhecidas

Como vimos existe atualmente muitas ferramentas de ETL, e algumas são mais populares e outras nem tanto, por isso vamos citar algumas dessas que são mais populares.

A tabela a seguir fornece-nos uma lista das ferramentas de ETL existentes e mais conhecidas, e a respectiva versão e fornecedor de cada uma.

Nº.	Ferramenta ETL	Versão	Fornecedor
-----	----------------	--------	------------

1.	Oracle Warehouse Builder (OWB)	11gR1	Oracle
2.	Data Services	XI 4.0	SAP Business Objects
3.	IBM Information Server (Datastage)	8.1	IBM
4.	SAS Data Integration Studio	4.4	SAS Institute
5.	PowerCenter Informatica	9.5	Informatica
6.	Elixir Repertoire	7.2.2	Elixir
7.	Data Migrator	7.7	Information Builders
8.	SQL Server Integration Services	10	Microsoft
9.	Talend Studio for Data Integration	5.2	Talend
10.	DataFlow Manager	6.5	Pitney Bowes Business Insight
11.	Pervasive Data Integrator	10.0	Pervasive Software
12.	Open Text Integration Center	7.1	Open Text
13.	Oracle Data Integrator (ODI)	11.1.1.5	Oracle
14.	Data Manager/Decision Stream	8.2	IBM (Cognos)
15.	Clover ETL	3.1.2	Javlin
16.	Centerprise	5.0	Astera
17.	DB2 Infosphere Warehouse Edition	9.1	IBM
18.	Pentaho Data Integration	4.1	Pentaho
19.	Adeptia Integration Suite	5.1	Adeptia
20.	DMExpress	5.5	Syncsort
21.	Expressor Data Integration	3.7	QlikTech

Tabela 5 - Lista das Ferramentas ETL mais conhecidos¹⁰

A Figura 12 mostra o Quadrante Mágico da pesquisa feita pela Gartner¹¹ em Outubro de 2012 sobre as ferramentas de Integração de Dados. Pelo que podemos ver a Informática é o líder do mercado ficando no topo do quadro, e o Talend, que é a ferramenta que

¹⁰ Disponível em: <http://www.etltool.com/list-of-etl-tools/>, consultado em a 08 de Junho de 2013

¹¹ Gartner oferece pesquisa de tecnologia para líderes de negócios globais de tecnologia para tomar decisões informadas sobre iniciativas importantes.

vamos utilizar, como o visionário, e é a única ferramenta *OpenSource* no quadrante mágico de 2012.

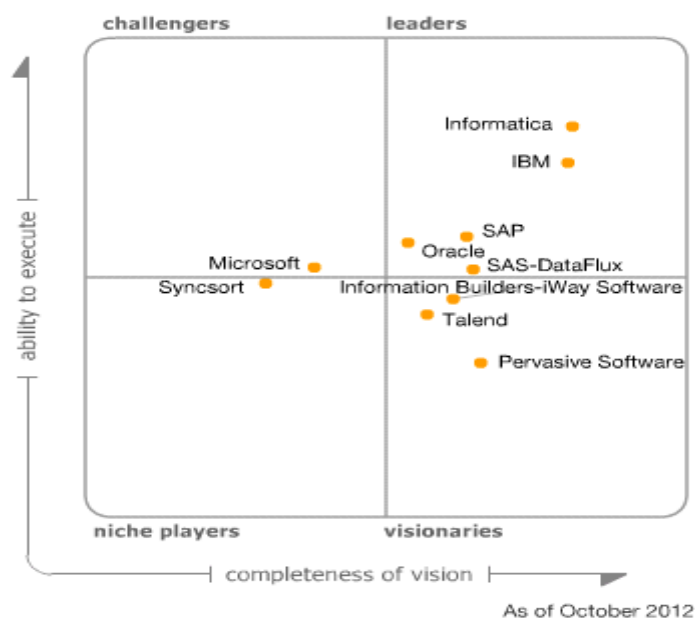


Figura 12 - Quadrante Mágico para Ferramenta de ID 2012

Fonte: (Thoo *et al*, 2012)

Mas existem ainda muitas outras soluções *OpenSource* de ETL, com qual vamos falar na seção seguinte.

Segundo Thoo *et al* (2012), o mercado das ferramentas de ID foi de 1,9 bilhões de dólares no final do ano 2011, com um aumento de 15,3% a partir de 2010. A taxa projetada de 5 anos de crescimento anual é cerca de 9% que trará um total de mais de 2,8 bilhões de dólares até 2016.

Na perspectiva de Thoo *et al* (2012), o mercado de ferramentas de ID, vem apresentando novas ofertas com mais desempenhos e escalabilidades necessárias para a integração de dados em escala empresarial. E os compradores deste mercado aumentaram a utilização dessas ferramentas e procuram novas tecnologias para atender a uma grande variedade de ID.

4.2 As *OpenSource*

Nesta seção vamos falar das ferramentas *OpenSource* de ID, mas antes vamos definir o que são as ferramentas *OpenSource*.

4.2.1 O que significa *OpenSource*

A OSI criou a o termo *OpenSource* e diz que não significa apenas acesso ao código-fonte e define alguns dos pontos que um *software OpenSource* deve garantir¹²:

1. Distribuição livre
2. Código fonte
3. Trabalhos Derivados
4. Integridade do autor do código fonte
5. Não discriminação contra pessoas ou grupos
6. Não discriminação contra áreas de atuação
7. Distribuição da Licença
8. Licença não deve ser específica para um produto
9. Licença não deve restringir outros *Softwares*
10. Licença deve ser tecnologicamente neutra

¹² Disponível em: <http://opensource.org/docs/osd>, consultado a 09 de Junho de 2013

4.2.2 As Ferramentas de ETL OpenSource

Segundo Altos (2008), a primeira ferramenta *OpenSource* de ID que apareceu foi ***Talend Open Studio for Data Integration***, em 2006 fornecida pela Talend com a licença GPL v2.

Desde então muitas ferramentas *OpenSource* apareceram, como podemos ver, algumas dessas na tabela a seguir. (Altos, 2008)

Nº.	Ferramenta ETL	Fornecedor
1.	Talend Open Studio	Talend
2.	KETL	Kinetic Networks
3.	CloverETL	Javlin
4.	Pentaho Kettle	Pentaho
5.	Apatar	Apatar, Inc
7.	Jitterbit	Jitterbit
8.	DataCleaner	eobjects.org

Tabela 6 - Ferramenta OpenSource para ID¹³

Pode-se encontrar outras ferramentas *OpenSource* também disponíveis no mercado. Mas nessa tabela nós optamos por distinguir algumas delas.

O Talend lidera o mercado de ferramentas *OpenSource* de integração de dados e segundo o relatório apresentado por Yuhanna (2012), fez por merecer o lugar na categoria de líderes oferecendo um forte apoio para as opções de implementação, técnicas de integração e conectividade com várias fontes de dados.

Mais informações sobre essa ferramenta Talend na seção a seguir, pois essa é a ferramenta que será utilizada para o caso prático que vamos apresentar no próximo capítulo.

¹³ A tabela foi criada baseada nas ferramentas *opensource* de Integração de dados existentes

5 Comprar ou Desenvolver?

No início da criação de um projeto de DW, as empresas tem que tomar uma decisão: se vai comprar ou se vai desenvolver a sua própria ferramenta ETL. Mas essa decisão não é assim tão fácil como parece pois existem muitos aspectos a considerar antes da tomada de qualquer decisão.

Segundo DWH (2011), quando se trata de seleção de uma ferramenta ETL nem sempre é necessário comprarmos uma ferramenta de terceiros. Mas também não vamos diretamente começar a desenvolver sem nenhum estudo ou avaliação.

Na construção de qualquer projeto de DW, precisamos de uma solução ETL, seja ela um ferramenta comprada ou desenvolvida manualmente por programadores interno.

Quando chegar a hora de comprar ou desenvolver, existem alguns aspectos que podem nos guiar para uma melhor decisão. Segundo Madsen (2004), estes aspetos podem ser:

- **Custo**

Com a crise que assola o mundo, o preço de muitos produtos diminuíram, mas o custo dos produtos de ETL aumentaram nos últimos anos, apesar de uma tendência de ligeira diminuição no ano anterior (2012) segundo Thoo, Friedman, & Beyer (2012), na sua última pesquisa sobre as ferramentas de integração de dados. Os fornecedores driblam essa crise aumentando constantemente as funcionalidades, desempenhos e usabilidades a cada nova versão (Madsen, 2004).

Quanto ao custo podemos ter dois grupos de produtos (Madsen, 2004): os ETL suites (que possuem preços elevados e mais bem orientados) e os produtos de ETL customizados consideravelmente menos.

O importante é sempre avaliar o projeto de uma forma realista, pois muitos projetos usam produtos caros sem nem mesmo analisar (considerar) outros produtos que são de baixo custo (baixa qualidade), ou codificação manual, se estes não seriam mais fáceis, mais práticos ou ainda mais rentáveis (Madsen, 2004).

- **Staffing** (Pessoal)

Se vamos optar por desenvolver (escrever códigos) nosso próprio ETL, é imperativo termos programadores qualificados para desenvolver o projeto. Esse pessoal tem que ter alguns requisitos como a familiaridade com o sistema DW operacional, Base de Dados, e fontes de dados, ser experiente na linguagem que escolhemos, ter grandes habilidades com SQL e possuir pelo menos um conhecimento básico de Data warehousing (Madsen, 2004).

Caso tivermos desenvolvedores com capacidades, podemos construir nosso ETL, mas caso contrário, é melhor comprarmos para não sofrer grandes consequências.

- **Fontes de dados heterogénios**

É comum num projeto encontrar fontes de dados heterogénios. Mas as ferramentas normalmente favorecem muito quando se fala de fontes de dados (Madsen, 2004).

Quando codificamos ETL à mão, é necessário ter desenvolvedores com habilidades adicionais para lidar com todas as plataformas de fonte de dados (Madsen, 2004).

- **Pacote de Software como fonte primária**

Normalmente nas organizações, podemos encontrar como fonte de dados alguns pacotes de software, e caso não possuamos um conector a essa fonte, temos que a criar, o que exigirá um grande esforço.

Segundo Madsen (2004), quando pacotes de software, tais como sistemas ERP são fonte de dados, um bom protudo de ETL pode melhorar muito a produtividade do desenvolvedor. Isso porque não são obrigados a estudar e desenvolver um conector para esses pacotes de softwares.

A maioria dos principais fornecedores de ETL possuem interfaces de extração pré-constituído para pacotes de software e geralmente essas interfaces possuem um conector

para a fonte de dados para que o desenvolvedor tenha o mínimo de trabalho na configuração para obter os dados pretendidos (Madsen, 2004).

Madsen (2004) diz ainda que mapear a extração de dados nos pacotes é muito difícil a não ser que sejamos um especialista nos detalhes de implementação subjacente.

É preciso aprender o *schema* do fornecedor e as regras de processamento através de sua documentação e também podemos precisar de alguns componentes da engenharia reversa para entender como eles funcionam (Madsen, 2004).

Como o pacote é sempre atualizado, alerta Madsen (2004), que é necessário criar um mecanismo de detecção dessas mudanças e gerenciar as mudanças de ETL.

- **Alterações frequente das fontes**

Alterações nos aplicativos internos da organização pode não ser tão bem coordenados ou controlados e quando uma organização enfrentar frequentes mudanças no sistema fonte, então um produto ETL será a melhor escolha do que construir seu próprio produto de zero (Madsen, 2004).

Quando há uma alta rotatividade do pessoal, a manutenção do ETL a longo prazo pode ser um grande problema caso o código for desenvolvido pelo pessoal interno, e isso faz com que a produtividade de novas pessoas seja muito menor, pois esses novos funcionários tem que entender tanto o subsistema de ETL em que vão trabalhar e o sistema de origem de dados (Madsen, 2004).

Por outro lado, os novos funcionários podem ser treinados rapidamente em uma ferramenta ETL, acrescenta Madsen (2004).

A manutenção em longo prazo é uma das áreas que pode ser utilizado para ajudar a justificar a compra do produto em vez da construção, principalmente quando os sistemas de origem estão num estado de fluxo constante (Madsen, 2004). Por outro lado, se os sistemas são relativamente estáveis e não há planos para grandes

atualizações num futuro próximo, construir ETL pode ser uma opção rentável (Madsen, 2004).

- **Fontes personalizadas**

Grande parte do esforço de desenvolvimento de ETL é gasto no mapeamento e análise dos dados. Esse mapeamento inclui a identificação das fontes mais confiáveis, identificar dados ruins, lidar com exceções de dados, entre outros. Todas essas tarefas devem ser concluídas através da ferramenta ETL ou não (Madsen, 2004).

Se a fonte de dados povem de aplicações personalizados, nesse caso, uma ferramenta ETL não oferecerá um início mais rápido do que a construção de ETL (Madsen, 2004). Isso também acontece se a ferramenta ETL não tem uma extração pre-construída para um pacote de software que estamos a utilizar como fonte de dados, acrescenta Madsen (2004).

As vezes, podemos passar muito tempo a tentar desenvolver uma solução que já se encontra disponível, por isso, Madsen (2004) diz que devemos olhar para o que estamos a construir, com o que os fornecedores fornecem.

“A resposta certa sobre comprar ou construir depende do seu orçamento, cronogramas do projeto e as características que são importantes para a sua aplicação específica.” (Madsen, 2004).

De acordo com DWH (2011), temos de considerar outros três aspetos fundamentais antes de tomar qualquer decisão:

1. **A complexidade da transformação de dados** – quanto mais complexa for a transformação dos dados mais adequados é comprar uma ferramenta ETL.
2. **Necessidade de limpeza de dados** – será que os dados precisam passar por uma limpeza profunda antes de serem armazenados no DW? Se for a melhor solução será comprar uma ferramenta com uma grande capacidade de

limpeza de dados. Caso contrário, pode ser suficiente a construção de uma rotina ETL a partir de zero.

3. **Volume de dados** – qual será o volume de dados a serem extraídos? Todos esses dados vão passar por uma transformação e limpeza? Quando maior é o volume de dados menor é o desempenho na transformação e na limpeza dos dados, aumentando assim o tempo do processamento. Ferramentas comerciais normalmente possuem recursos para acelerar a circulação de dados. Nesse caso se o volume de dados for muito grande, a melhor solução é comprar.

Por as vezes por necessidade somos obrigados a seleccionar um BD, uma plataforma de *hardware* ou ferramentas de BI, mas quando se trata da selecção de uma ferramenta de ETL, não somos obrigados mas é altamente recomendado que a tenhamos (DWH, 2011). Diz ainda Madsen (2004) que depois de comprar hardware, software de BD e ferramentas de BI, podemos não ter dinheiro suficiente no orçamento capital para comprar uma ferramenta ETL.

Difícilmente pode-se encontrar uma ferramenta com uma grande capacidade de transformação e limpeza (DWH, 2011), ou seja, é mais provável encontrarem um que tenha ou mais capacidade de limpeza ou mais capacidade de transformação.

Como cada organização possui fontes de dados diferentes, diz DWH (2011) que é necessário certificar antes de comprar, se essa ferramenta pode conectar directamente com todas as fontes de dados que existem naquela organização.

E Kimball *et al* (2008), uma ferramenta ETL pode melhorar a produtividade, mas isso não quer dizer que o fará em um dia. Pois é preciso ser um pouco paciente e não tentar fazer tudo da forma mais rápida, porque o importante é que o sistema funciona como esperado.

Vamos nos pontos a seguir definir as vantagens e desvantagens que possuem a codificação manual e a ferramenta ETL, para uma melhor compreensão e selecção.

5.1 Codificação Manual

A primeira solução de ETL que apareceu era uma forma de codificação manual. Mesmo com o surgimento das novas ferramentas e seu aperfeiçoamento, muitas organizações ainda utilizam a codificação manual para desenhar o seu sistema ETL, isso por razões adversas.

Se uma organização optar pela codificação do seu sistema pode conseguir fazer exatamente o que pretende, mas também pode correr o risco em relação a qualidade dos dados. Apesar de ainda ter que possuir uma equipa com grandes habilidades e especialidades.

Segundo Ponniah (2010), ninguém consegue construir totalmente um DW a partir de zero com a programação interna.

Vamos então citar algumas vantagens e desvantagens que possuem os que preferem codificar seu sistema.

➤ Vantagens

- Técnicas de programação orientadas a objeto ajudá-lo a fazer todas as suas transformações consistente para o relatório de erros, validação e atualização de metadados (Kimball & Caserta, 2004).
- Os metadados criados podem ser controlados de uma forma mais direta (Zode, 2011), embora ao mesmo tempo, deve-se criar todas as interfaces de metadados, acrescenta Kimball & Caserta (2004).
- O teste do código ETL escrita é fácil, pois estão disponíveis muitas ferramentas de teste automatizadas, para sistemas codificados à mão (Zode, 2011).

- Codificação manual é mais flexível, pode-se fazer o que quiser sem qualquer limitação (Zode, 2011). Na mesma perspectiva Kimball & Caserta (2004), diz ainda que, em muitos casos uma abordagem única ou uma linguagem diferente pode proporcionar uma grande vantagem.

➤ **Desvantagens**

- Possuem uma menor velocidade de execução, pois os programas gerados manualmente são *threads* simples (Zode, 2011).
- Metadados é o suporte principal para qualquer projeto de DW, por isso codificação manual de ETL requer que manutenção de metadados seja feita separadamente, pois cada mudança requer alteração na tabela de metadados e isso é feito manualmente (Zode, 2011).
- A manutenção desses Sistemas de ETL é um grande problema segundo Zode (2011).

5.2 Ferramentas de ETL

O uso das ferramentas de ETL vem facilitando as organizações, principalmente na qualidade dos dados e na sua rapidez.

Podemos ver, de acordo com o percurso de ETL, que as ferramentas de ETL evoluíram e atualmente ela proporciona uma grande variedade de transformações, maior desempenho, escalonamento, monitoramento, etc.

O uso das ferramentas nos torna muito limitado, uma vez que estamos sujeitos a fazer o que a ferramenta possui a capacidade. Mas as novas ferramentas possuem uma alternativa a esse problema, pois possui recursos onde é possível codificar o que desejamos e não é possível fazer com a ferramenta (Um exemplo disso é o Talend).

Segundo Mundy (2008), existem enumeras vantagens¹⁴ associadas a ferramentas ETL, por isso vamos citar algumas vantagens e desvantagens do uso dessas ferramentas.

➤ **Vantagens**

- **Fluxo visual e Auto documentação** – as ferramentas possuem uma interface onde é possível visualizar o fluxo criado, mas cada ferramenta possui um visual diferente. Também possui a capacidade de auto documentar o programa criado (Mundy, 2008);
- Técnicos que não sejam programadores podem utilizar a ferramenta de ETL sem problemas (Kimball & Caserta, 2004);
- **Projeto de sistema estruturado** - é particularmente valioso para as equipas construindo seu primeiro sistema de ETL (Mundy, 2008);
- Muitas ferramentas de ETL possuem repositório integrado de metadados que podem sincronizar metadados de sistemas de origem, BD de destino, e outras ferramentas de BI (Kimball & Caserta, 2004);
- Fornece funcionalidades e práticas para operar e monitorar um sistema de ETL em produção (Mundy, 2008);
- Ferramentas ETL possuem conectores pré-construídos para a maioria dos sistemas de origem e destino (Kimball & Caserta, 2004);
- **Limpeza de dados avançada** – como as fontes de dados são muitas vezes heterogenias, muitas vezes é necessário uma ferramenta para fazer a limpeza desses dados (Mundy, 2008);

¹⁴ Podemos encontrar mais vantagens também no Kimball & Ross (2010, pp. 442,443) e Kimball & Caserta (2004).

- **Desempenho** – desempenho das ferramentas de ETL é muito elevado mesmo quando o volume de dados é muito grande (Kimball & Caserta, 2004).

➤ Desvantagens

- **Custo de licença** – o custo de uma ferramenta de ETL é muito elevado (Mundy, 2008).
- **Incerteza** – segundo Mundy (2008), muitas equipas de ETL não sabem o que uma ferramenta pode fazer por eles.
- **Flexibilidade Reduzida** – limita a capacidade o fornecedor e das linguagens de *script*¹⁵.

6 A Ferramenta Talend Open Studio para Integração de Dados

A ferramenta Talend Open Studio para ID é um ambiente gráfico *OpenSource* de desenvolvimento para criação e *deploy* de uma integração personalizada entre os sistemas (Bowen, 2012).

Segundo Altos (2008), Talend foi o primeiro a fornecer software *OpenSource* para integração de dados.

E na perspetiva de Bowen (2012), essa ferramenta possui mais de 600 conectores pré-construídos que o torna mais rápido e fácil a conexão com BD, transformações de ficheiros, *load* de dados, copiar e renomear ficheiros e conectar componentes individuais a fim de definir um processo de integração complexo.

Talend Open Studio para ID é um gerador de código (Bowen, 2012), e faz com que muitos trabalhos difíceis se tornam simples. Também, diz Bowen (2012), é uma

¹⁵ Linguagens de Script - são linguagens de programação executadas do interior de programas e/ou de outras linguagens de programação, não se restringindo a esses ambientes. Disponível em: http://pt.wikipedia.org/wiki/Linguagem_de_script, consultado a 11 de Agosto de 2013

ferramenta fácil de usar mesmo para desenvolvedores experientes e pessoas não desenvolvedores.

A figura a seguir mostra a interface da ferramenta *Talend Open Studio for Data Integration* versão 5.3.0.

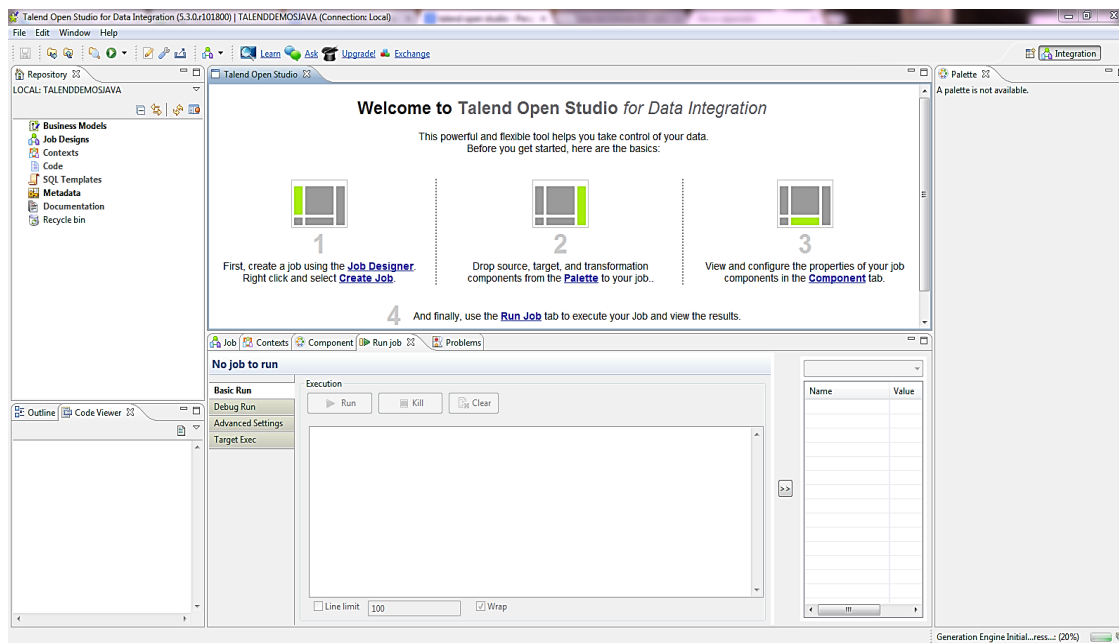


Figura 13 - Talend Open Studio for Data Integration

No Talend é possível a criação de *Jobs* e configurar componentes em vez de codificar, e esses *Jobs* podem ser executados a partir do ambiente gráfico de desenvolvimento ou executado como scripts independentes (Bowen, 2012). Também na execução é possível ver o fluxo, e também possui uma forma de *debug*.

Talend funciona em varias plataformas como Windows, Linux e Mac.

Talend tem uma grande comunidade onde dificilmente não se encontra quem não tenha algum problema igual a o que podemos deparar no desenvolvimento do nosso sistema.

6.1 História

Talend foi fundada em 2005, e é fornecedor de *software OpenSource* fornecendo soluções para integração de dados, qualidade de dados, MDM, ESB e BPM (Bowen, 2012).

A figura a seguir mostra um histograma da companhia Talend.

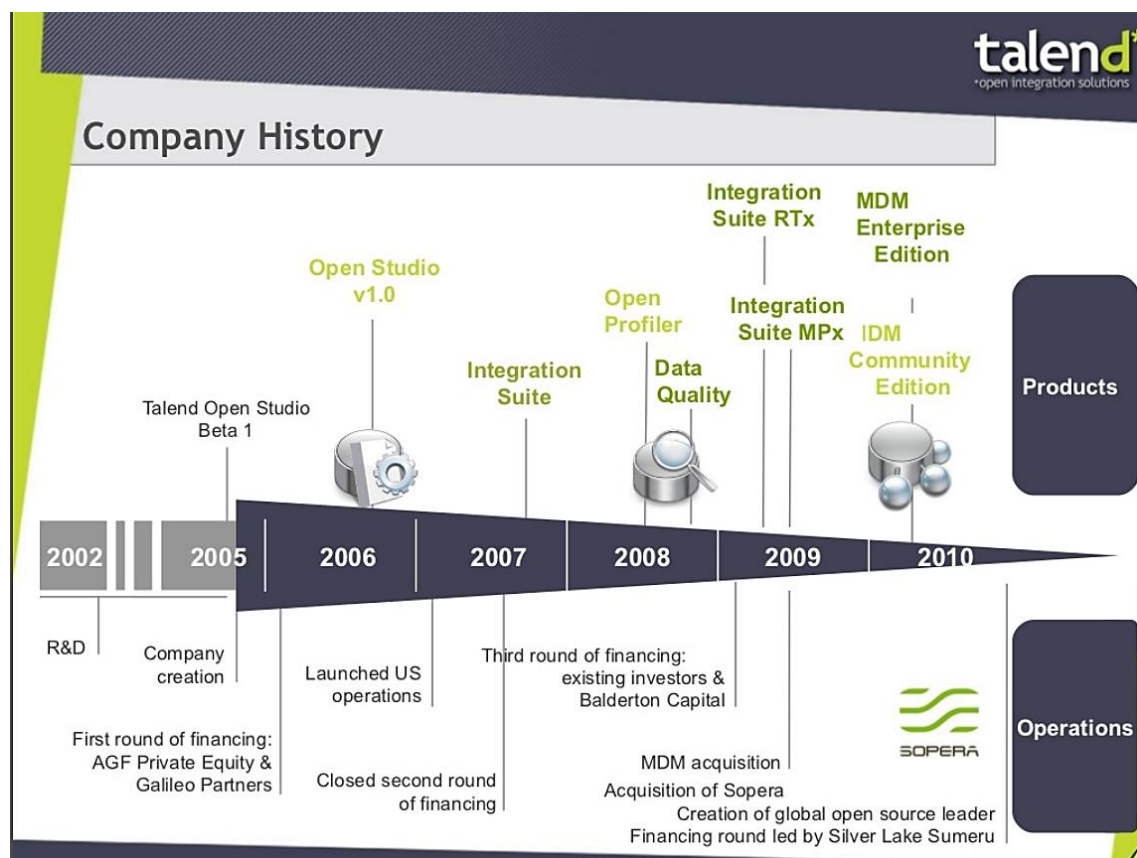


Figura 14 - História da companhia Talend¹⁶

Segundo Bowen (2012), o primeiro produto de Talend, TOSDI, foi lançado em 2006, com o nome, *Talend Open Studio*, e desde então já foi descarregado mais de 20 milhões de vezes.

Segundo Brodtkin (2007), a primeira versão lançada foi 2.0, com a intenção de facilitar o uso das ferramentas de integração de dados a pequenas e médias empresas que não

¹⁶ Disponível em: <http://www.slideshare.net/industrialtsi/talend-intruccion-by-tsi>, pág. 4, consultado a 2013-06-19

têm como pagar as ferramentas de fornecedores de software proprietário, como Informatica, Oracle ou IBM.

Ainda Talend desenvolve esse produto e a versão atual é a 5.3, uma versão com muito mais melhorias.

6.2 Benefícios

Talend Open Studio for Data Integration proporciona um conjunto de benefícios para quem optar pelo seu uso na construção de um projeto.

Segundo Bowen (2012), essa ferramenta fornece os seguintes benefícios:

- É *OpenSource*, livre para download e uso, com acesso a código fonte, permitindo aos utilizadores estender o produto de acordo com as suas necessidades específicas.
- É um grande impulsionador da produtividade. É fácil de aprender e rápido para desenvolvimento. Mesmo os desenvolvedores principiantes poderão construir integrações complexas de uma forma muito rápida.
- Usa componentes pré-construídos para auxiliar nas tarefas.
- Tem uma grande comunidade¹⁷ de utilizadores aberta e ativa.

¹⁷ Para aceder a comunidade da Talend aceda <http://www.talendforge.org/forum/>

Capítulo 3: Caso Prático – Uso da Ferramenta TOSDI para integração e controlo dos dados

1 Enquadramento

No decorrer deste Capítulo vai ser apresentado um caso prático sobre a integração de dados na empresa de Telecomunicação Unitel T+. E vamos fazer a apresentação da empresa em estudo, a sua organização, o percurso da ID, bem como identificar o problema com o sistema existente, a solução proposta, e quais foram as vantagens do sistema desenvolvido em relação ao existente.

Foram efectuadas entrevistas a 3 pessoas que já tinham trabalhado com o sistema antigo, para dar suporte à análise que vamos apresentar.

2 Apresentação da empresa

A empresa Unitel T+, antiga T+, com sede em Achada Santo António, pertence ao grupo Unitel internacional, e é uma operadora de telefonia móvel em Cabo Verde inovadora na prestação de serviços de telecomunicações de qualidade que aposta fortemente na oferta de uma tecnologia simples e acessível para melhor responder as necessidades dos seus clientes, levando sempre a melhor e mais completas soluções para o mercado da telefonia móvel.

A marca Unitel T+ tem como valores:

- **Qualidade**, coerência na comunicação e qualidade dos serviços;
- **Liderança**, fomentando a confiança dos seus clientes na Unitel T+ como líder em tecnologia, na satisfação dos nossos clientes e na comunicação;
- **Inovação**, no lançamento de novos produtos e serviços.

2.1 Organização Geral

A empresa Unitel T+ é constituída por 3 direcção:

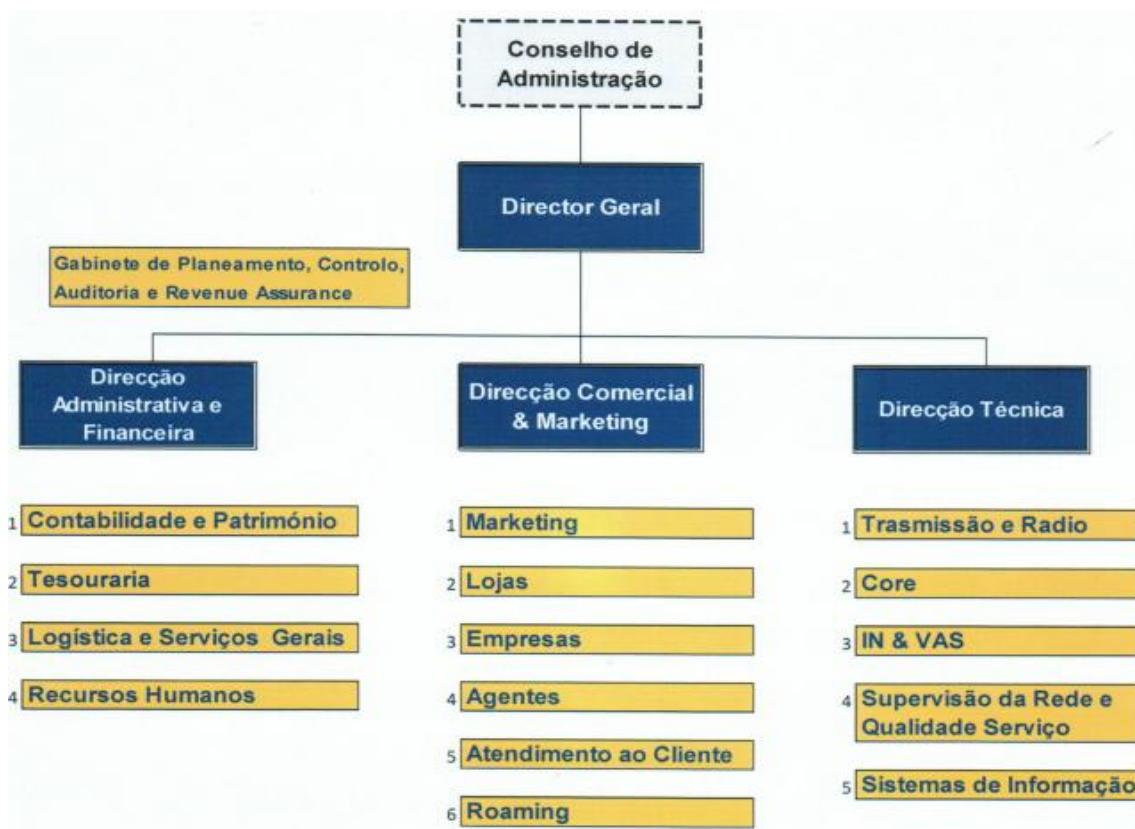


Figura 15 - Estrutura Organizacional da Unitel T+¹⁸

Como o projeto foi desenvolvido para o departamento técnico na área IT, vamos identificar algumas das funções desempenhadas na área IT:

¹⁸ Estrutura enviada pela RH da empresa

- ❖ Desenvolvimento (Integração de Aplicações, Módulos para portal interno)
- ❖ *Reporting* (Integração de Dados, ETL, *Reports*)
- ❖ Infra-estrutura de IT (Sistemas, Virtualização, Rede e Segurança)

3 Integração de Dados na Unitel T+

Como vimos no Capítulo 2 a ID ajuda a integrar os dados que se encontram espalhados pela empresa, tendo assim um maior controlo desses dados.

Como na Unitel T+ trabalha-se com uma grande quantidade de dados, é imperativo ter um sistema para integrar esses dados.

3.1 Histórico dos métodos ou ferramentas utilizadas para ID

Em 2007 a empresa T+ Telecomunicações, surgiu como uma nova operadora de telefonia móvel em Cabo Verde, e em Dezembro do mesmo ano já contava com cerca de 3308 clientes¹⁹. A tendência era sempre ter mais números de clientes possíveis, e quanto mais o número de cliente, mais será os dados para processamento.

Inicialmente esse sistema de GSM era de ERIKSON, mas com a implementação de 3G, passou para HUAWEI, logo a geração dos ficheiros alteraram.

Esses ficheiros são chamados de CDR²⁰, e são gerados de acordo com o fluxo da chamada:

- **Incoming** – chamadas recebidas pelos clientes da Unitel T+ (CDR- MTC, GWI, TERMCAMEL);
- **Outgoing** – chamadas efectuadas pelos clientes da Unitel T+ (CDR – MOC, GWO, CFW).

¹⁹ Essa informação foi retirada de um relatório mensal que era enviado para ANAC

²⁰ CDR são gerados no MSC e colocados no FTP

Existe ainda as chamadas de Roaming com CDR gerado como ROAM. Também temos os CDR de CBS, que foram os utilizados na construção do modelo de sistema ETL.

Nessa empresa a integração de dados passou por 2 etapas que não poderíamos deixar de descrever, são eles:

- **Codificação Manual**
- **Uso de Ferramentas de ETL**

3.1.1 Codificação Manual

Essa foi a primeira metodologia utilizada na empresa para processar os dados. Apesar de já existirem ferramentas que pudessem facilitar esse trabalho, a equipa optou por codificar. Todo o trabalho era “quase” feito manualmente, como a extração do ficheiro e o controlo desses ficheiros.

A figura a seguir mostra como era o sistema que era utilizado no início.

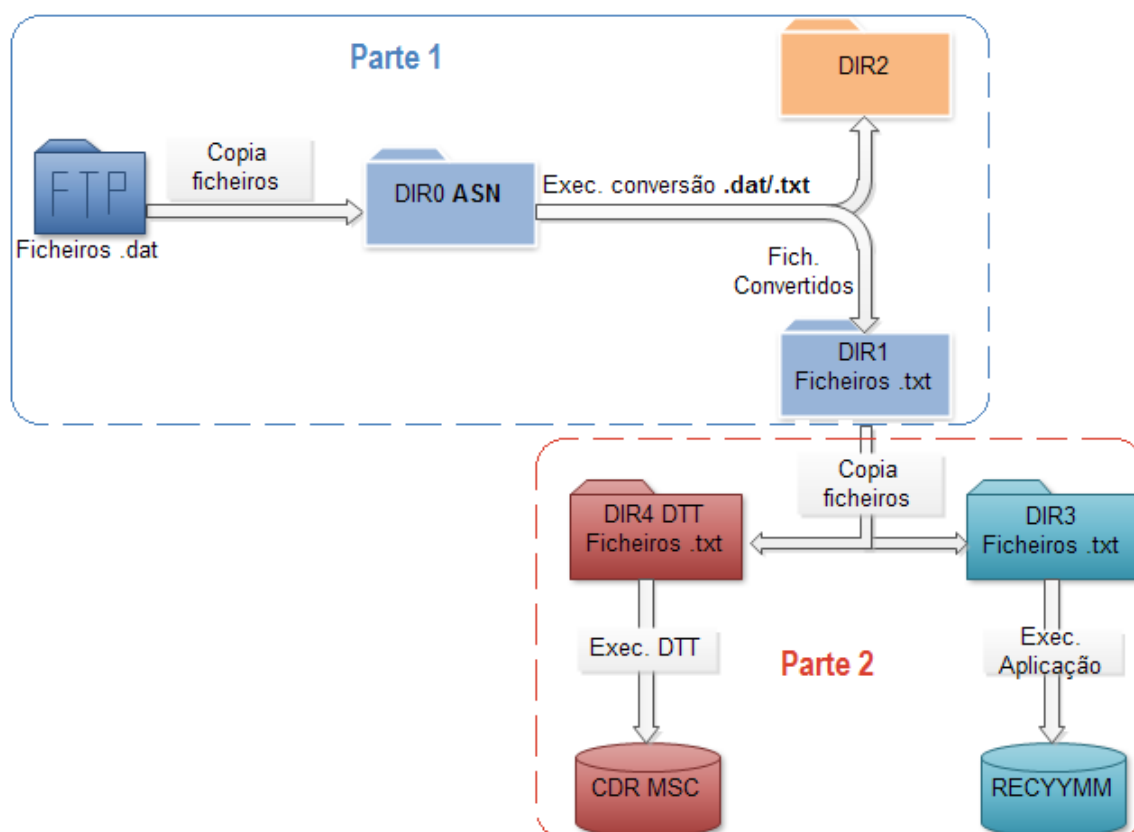


Figura 16 - Processamento dos ficheiros no início²¹

Segundo a Figura anterior na “Parte 1”, os ficheiros que eram gerados na MSC se encontravam num FTP, depois esses ficheiros (todos) eram copiados para a diretoria ASN (DIR0). Na altura tinham uma aplicação²² (.exe) que para cada ficheiro copiado, essa aplicação iria converter esse ficheiro do formato ASN para texto (.txt) e colocar na diretoria (DIR1), mas caso esse ficheiro já tinha sido convertido, seria colocado numa outra diretoria (DIR2).

Até aqui foi feito a preparação dos ficheiros texto, que na “Parte 2” serão copiados para 2 diretorias, sendo na primeira diretoria (DIR3), será executado uma aplicação para inserir os dados desses ficheiros na tabela de acordo com o ano e o mês que esse ficheiro pertence, usando o modelo RECYYMM. Por exemplo, para um ficheiro de Maio de 2011, o nome da tabela seria REV1105. Na segunda diretoria (DIR4 DTT),

²¹ Criado a partir de uma documentação do sistema antigo da Unitel T+

²² Essa aplicação chamava-se ConvertASN_New que convertia os ficheiros ASN1 para Texto

seria executado uma aplicação chamada de DTT (Data Transform Tools- codificado em java), que fazia a inserção de todos os ficheiros na tabela CDR_MSC²³.

O controlo era feito depois de inseridos os dados nas tabelas, por isso criaram um ficheiro Excel (.xls), que continha as informações a cerca dos ficheiros convertidos e processados. A figura a seguir mostra um exemplo desse ficheiro.

FILESYSTEM		BD		DATE	First	Last	Nr Processed
FIRST	LAST	FIRST	LAST				
CIBCERIC_CPVTM_110106	CIBCERIC_CPVTM_110189	cdr_11_110106	cdr_11_110189	25-11-2011	0106	0189	84
				26-11-2011			
				27-11-2011			
CIBCERIC_CPVTM_110190	CIBCERIC_CPVTM_110434	cdr_11_110190	cdr_11_110434	28-11-2011	0190	0434	245
CIBCERIC_CPVTM_110435	CIBCERIC_CPVTM_110524	cdr_11_110435	cdr_11_110524	29-11-2011	0435	0524	90
CIBCERIC_CPVTM_110525	CIBCERIC_CPVTM_110610	cdr_11_110525	cdr_11_110610	30-11-2011	0525	0610	86
CIBCERIC_CPVTM_110611	CIBCERIC_CPVTM_110687	cdr_11_110611	cdr_11_110687	01-12-2011	0611	0687	77
CIBCERIC_CPVTM_120688	CIBCERIC_CPVTM_120719	cdr_11_120688	cdr_11_120719	02-12-2011	0688	0719	32
				03-12-2011			
				04-12-2011			
CIBCERIC_CPVTM_120720	CIBCERIC_CPVTM_121073	cdr_11_120720	cdr_11_121073	05-12-2011	0720	1073	354
CIBCERIC_CPVTM_121074	CIBCERIC_CPVTM_121161	cdr_11_121074	cdr_11_121161	06-12-2011	1074	1161	88
CIBCERIC_CPVTM_121162	CIBCERIC_CPVTM_121262	cdr_11_121162	cdr_11_121262	07-12-2011	1162	1262	101

Figura 17 - Controlo inicial dos ficheiros²⁴:

Para verificação de omissão dos dados e, era utilizado 2 Querys²⁵:

- O primeiro para certificar que os ficheiros inseridos na tabela são iguais aos que foram convertidos. Ex:

```
select count (distinct filename)
FROM [ERICSHDB].[dbo].[CDR_MSC]
where filename >= 'cdr_10_085201.txt'
and filename <= 'cdr_10_085275.txt'
```

- O segundo verificar se todos os registos foram inseridos. Ex:

```
SELECT cast(SEQUENCE as int) as
recordsequencenumber, FILENAME, DATE_CALL
FROM [ERICSHDB_REV].[dbo].[REV1010]
where FILENAME >= 'cdr_10_108154.txt' and
FILENAME <= 'cdr_10_108220.txt'
order by [recordsequencenumber]
```

²³ Tabela que contem todos os ficheiros (CDR's).

²⁴ Retirado do ficheiro Excel de controlo dos ficheiros que eram inicialmente utilizados

²⁵ Retirado da documentação do projeto

Segundo os dados da entrevista, um dos grandes problemas dessa metodologia era a duplicação de dados e nulos, pois se houver algum erro no processamento dos ficheiros e quando vão ser reprocessados não existe nenhum controlo para evitar a duplicação dos dados. Também a aplicação que fazia a conversão dos ficheiros em texto retornava muitos valores nulos, e não existia também um controlo dos valores nulos.

Também como os dados eram todos extraídos (copiados), tornava o processo mais lento, pois mesmo os ficheiros que já foram processados seriam copiados porque não existia um controlo nesse ponto, e isso aumentava o tráfego na rede.

3.1.2 *Uso de Ferramentas ETL*

Com a explosão das ferramentas *OpenSource* de ID no mercado, a Unitel T+ não hesitou em adoptar novas tecnologias, por isso em 2007 introduziu a primeira ferramenta de ETL para a ID, e foi o JasperETL com a versão 2.0.

Após a avaliação entre várias soluções afins tais como a Kettle, JasperETL e *SQL Server Business Intelligence Development Studio (SSIS)*, a Unitel T+ Telecomunicações adoptou-o para projectos relacionados com tratamento de dados entre outros propósitos.

E o JasperETL foi substituído pelo Talend Open Studio com versão 4.0.2, pois esse fornecia mais componentes para o desenvolvimento das tarefas (Segundo os entrevistados). Possuía também maior número de conectores para BD. Desde então até hoje a empresa utiliza essa ferramenta para ID.

Com o Talend os ficheiros eram extraídos a partir de um FTP, onde se encontrava os diferentes ficheiros (Data, Rec, SMS, MMS, etc). Essas fontes de dados serão mencionadas nas próximas seções.

A figura que se segue ilustra a árvore dos ficheiros no FTP. Podemos ver que a diretoria principal é “cbs”, e dentro dela se encontra outras diretorias denominadas com os nomes dos diferentes ficheiros (Rec, Data, SMS, MMS), e dentro de cada uma destas diretorias encontramos outras diretorias com nomes referentes a cada dia no formato

“YYYYMMDD”(Ex. para ficheiros de 10 de Agosto de 2013, o nome da directoria seria “20130810”), e dentro de cada um destas diretorias podemos encontrar os ficheiros.

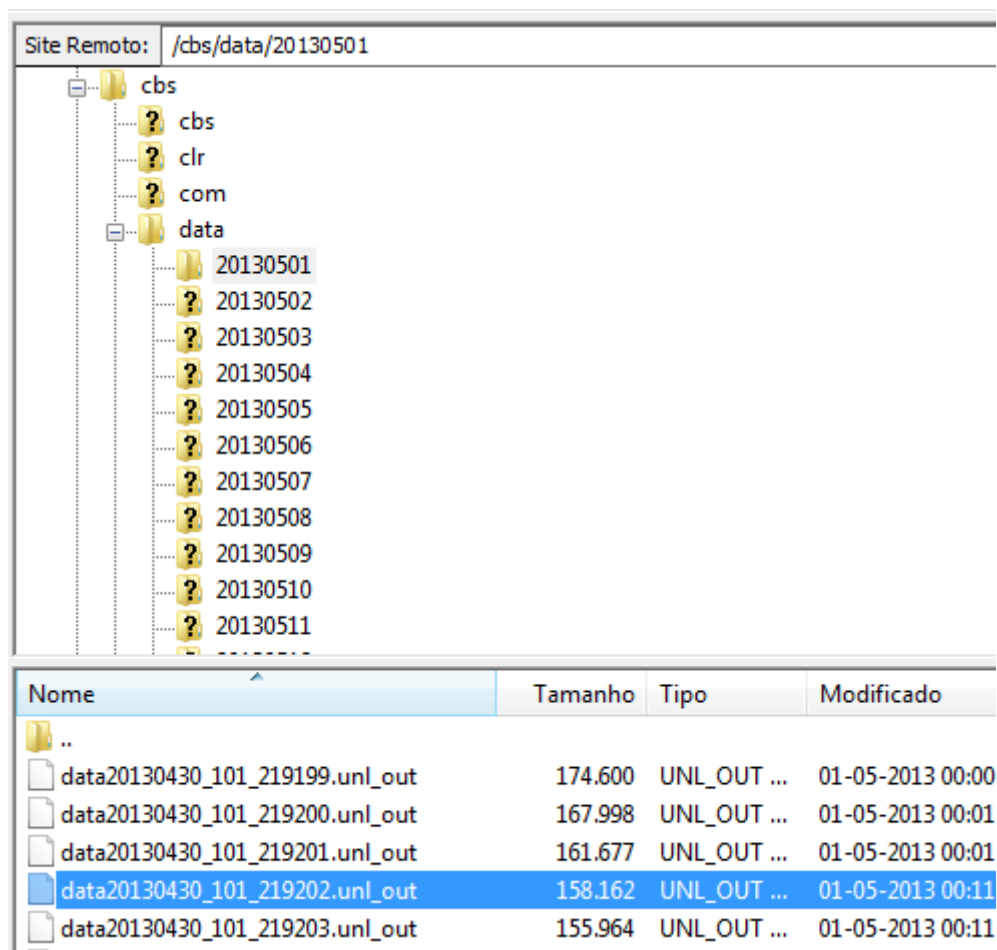


Figura 18 – Árvore FTP

3.2 O Problema

Inicialmente, como vimos, tudo era feito à mão, o que fazia com que esse processo estivesse sujeito a erros ou incoerência dos dados.

De acordo com os dados da entrevista o processo antigo poderia demorar Horas para terminar um processamento de um dia de dados, que com o novo sistema esse pode chegar no máximo 5 minutos.

Mas passando algum tempo, a empresa optou por incluir novas tecnologias e otimizar esse processo.

Mas mesmo com uso de ferramentas ETL (Talend Open Studio), ainda existia problemas no controlo dos ficheiros e dos dados que correspondem a cada ficheiro. Era normal encontrar dados duplicados e/ou falta de dados. Como não havia um controlo dos ficheiros inseridos, não se sabia a que ficheiro pertencia cada linha de dados na tabela.

De acordo com a observação feita, constatou-se que o sistema que existia não fornecia um controlo (mesmo com TOSDI), ou seja, o controlo que existia não funcionava. E isso não era bom, pois quando há falta de dados ou dados duplicados o problema era identificar o ficheiro a que esses dados pertenciam.

O outro problema era que não tínhamos como saber se esses dados estavam em faltas (Omissão de dados) ou duplicados, a não ser fazendo query. E como os relatórios eram feitos diários, se os dados estivessem com problemas, esse relatório com certeza não estaria perfeito.

Com BD em produção e com cerca de trezentos Gigabyte (300 GB) de dados, há duas opções para solucionar o problema dos dados em falta e dos duplicados:

1. No caso dos dados duplicados – apagar somente os duplicados, ou apagar todos e voltar a inserir.
2. No caso dos dados em falta – apagar os dados existentes e inserir todos, ou inserir só os que faltam.

Mas tudo isso nos leva a perguntar, de qual ficheiro são esses dados? Em qual ficheiro vamos procurar? Como fazer para inserir só os dados em falta? Essas são as questões que não podíamos evitar com esse sistema sem controlo.

Também um dos grandes problemas foi o aumento do tráfego na rede, pois como não havia um controlo na extração dos dados, esses seriam extraídos mesmo que já tinham sido inserido com sucesso na BD.

Na tabela sobre a comparação entre o sistema antigo e o atual, pode-se conferir os pontos menos positivos do sistema antigo.

3.3 A solução

Perante os problemas supracitadas, a solução foi a criação de duas tabelas relacionadas, em que na primeira tabela serão armazenadas as informações a cerca do ficheiro (data da inserção, nome de ficheiro, números de linhas, etc) e na segunda os dados que correspondem a esse ficheiro.

A figura a seguir mostra o relacionamento que existe entre as duas tabelas a serem criadas. Não foram adicionados todos os atributos, mas nas próximas seções vamos falar sobre isso.



Figura 19 – Relacionamento entre tabela de controlo e de dados

Com as tabelas criadas e com os seus respectivos relacionamentos, cria-se um processo ETL que tenha um controlo na extração e na inserção dos dados na BD, em que:

- O controlo na extração consiste em ir ao FTP e extrair somente os ficheiros não inseridos, isso diminui o tráfego na rede.
- O controlo na inserção consiste em inserir o ficheiro na tabela de controlo, mas usando um “*flag*”, que diz se o ficheiro foi ou não inserido com sucesso, isso permite que não haja dados duplicados e nem omitidos.

A figura a seguir ilustra como é feito esse controlo.

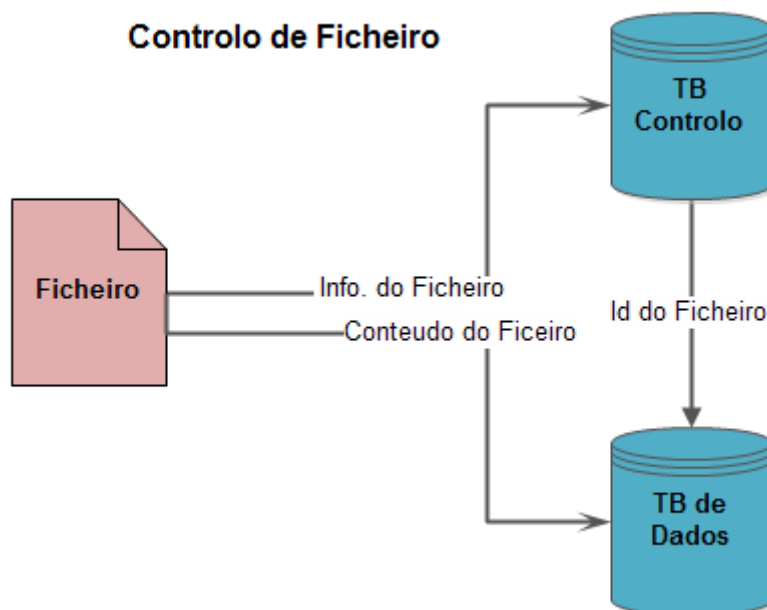


Figura 20 – Controlo dos ficheiros

Para isso foi criado um modelo de sistema para suportar todos os tipos de ficheiros que antes era processado mas sem controlo. Como cada ficheiro contém *schemas* diferentes a melhor solução era criar um projeto para cada um.

Na próxima seção vamos falar desses diferentes tipos de ficheiros que são fontes de dados.

4 O sistema desenvolvido

Dado o volume de dados inseridos diariamente (cerca de um milhão) torna-se necessário a existência de um sistema automático de controlo do fluxo dos dados a serem inseridos, nomeadamente evitar replicação e omissão de dados. Para tal, coube-nos a parte em que consistia no uso de metadados para o controle desse fluxo.

Uma vez que existem vários tipos de dados a serem extraídos e carregados para a base de dados, começamos com um modelo inicial para os CDR (ficheiro) de Dados que veio a ser extensível aos restantes tipos de CDR, pois existem vários tipos destes que são processados diariamente e alocados a respectivo repositório no FTP.

Esses CDR's serão extraídos no respectivo repositório no FTP, passam por um processo de transformação, mas, o principal objectivo é ter o controlo de cada CDR extraído e carregados para base de dados.

Esse controlo abrange, ter duas tabelas, como referido anteriormente, uma de controlo (com o nome do ficheiro), uma de dados (com dados que se encontram dentro de cada ficheiros), onde para cada linha inserida na tabela de dados será referenciado a um ficheiro da tabela de controlo.

Esse sistema foi desenvolvido usando a ferramenta Talend Open Studio versão 5.0, e a Base de Dados utilizada foi o *SQL Server*.

4.1 As Fontes de Dados

Como tínhamos falado anteriormente, as fontes de dados são os CDR's, e existem diferentes tipos desses CDR's. O propósito desse projeto era criar um modelo que suportava cada um desses ficheiros.

Esses ficheiros se encontram num FTP e estão distribuídas de acordo com duas diretorias, uma para ficheiros MSC e a outra para ficheiros CBS²⁶.

Os ficheiros na diretoria MSC são os gerados pelo MSC que correspondem a todas as chamadas efectuadas e recebidas pelos clientes da empresa, como tínhamos referido nas seções anteriores.

Os ficheiros na diretoria CBS são os que contêm as informações referentes a serviços prestados ao cliente. Por isso esses são os que serão cobrados aos clientes. Esses ficheiros são:

- **Rec** – esses ficheiros contêm dados referentes a chamadas.
- **SMS** – esses ficheiros contêm dados referentes a SMS.

²⁶ Ficheiros CBS são os que contem os dados que serão cobrados a clientes.

- **Data** - esses ficheiros contêm dados referentes a Dados.
- **CLR** - Quando o ciclo de vida dos assinantes muda, *clr CDRs* grava as informações se a CBS limpa os saldos das contas ou recompensa os assinantes.
- **MMS** - esses ficheiros contêm dados referentes a MMS.
- **MON** – esses ficheiros contêm dados referentes a subscrições de serviços.
- **MGR**²⁷ - esses ficheiros contêm dados referentes a ajustes efectuados manualmente.
- **Vou** - esses ficheiros contêm dados referentes a recargas e empréstimo de saldo.

Para esse sistema vamos utilizar os ficheiros de Data para construir um modelo.

4.2 Tabelas criadas

Foi também utilizado como Base de Dados o SQL Server. E como tínhamos referido anteriormente temos que criar duas tabelas, um de controlo de ficheiros e outro de dados. Isso nos ajudará a ter um maior controlo de cada ficheiro inserido na Base de Dados.

Essas tabelas criadas são:

I. TB_Control_DATA

O nome da tabela de controlo varia de acordo com o ficheiro (Ex. para Rec seria TB_Control_Rec). Essa tabela contém informações a cerca dos ficheiros processados. O principal objetivo dessa tabela é saber se um ficheiro foi ou não inserido completamente na outra tabela que contém os dados referentes a cada ficheiro. Com isso evitamos a

²⁷ Esses ajustes podem ser reposição ou subtracção de saldo na conta de um cliente manualmente, ou também no caso de alguns serviços que não são necessárias subscrições, como Amore, Horoscopo, etc.

extração dos ficheiros que já foram processados com sucesso, e diminuindo o tráfego na rede interna.


	Column Name	Data Type	Allow Nulls
	id	bigint	<input type="checkbox"/>
	nameFile	varchar(64)	<input type="checkbox"/>
	seq	varchar(16)	<input checked="" type="checkbox"/>
	flag	int	<input checked="" type="checkbox"/>
	date_modify	datetime	<input checked="" type="checkbox"/>
	Total_File_Record	int	<input checked="" type="checkbox"/>
	Total_insert_Record	int	<input checked="" type="checkbox"/>

Figura 21 - TB_Control_DATA

Essa tabela possui os seguintes atributos:

- **Id** (*bigint*) – incrementa 1 valor a cada linha inserido;
- **nameFile** (*varchar*) – nome do ficheiro;
- **seq** (*varchar*) – a sequencia do ficheiro;
- **flag** (*int*) – é 1 caso os dados não foram inseridos com sucesso na tabela de dados, e 2 caso contrario;
- **date_modify** (*datetime*) – dia e hora do processamento;
- **Total_File_Record** (*int*) – total de registos no ficheiro;
- **Total_insert_Record** (*int*) – total de registos inseridos na tabela de Dados referentes a esse ficheiro.

II. TB_DATA

Também o nome da tabela de dados varia de acordo com o ficheiro (Ex: para Rec seria TB_Rec). Essa tabela contém os dados referentes a cada ficheiro. Mas para identificar a

que ficheiro da tabela **TB_Control_DATA** um registo na tabela de **TB_DATA** pertence, foi criada uma relação entre essas duas tabelas, onde o ID da tabela de controlo será igual ao File_Id da tabela de dados, ou seja File_Id é a chave estrangeira.

O total de registo para um File_Id qualquer tem que ser igual a total de linhas que um ficheiro na tabela de controlo tem para ID = File_Id.

O atributo de cada ficheiro é diferente visto que se trata de dados diferentes, mas vamos ver na figura a seguir o atributo da tabela de dados para ficheiros de Data.

Column	Db Column	Key	Type	DB Type	<input checked="" type="checkbox"/>	Date Pattern (Ctrl+Sp...	Length	Precis...	Def.
File_Id	File_Id	<input type="checkbox"/>	long	BIGINT	<input type="checkbox"/>				
SerialNo	SerialNo	<input type="checkbox"/>	Long	BIGINT	<input checked="" type="checkbox"/>		8	0	
SubSequence	SubSequence	<input type="checkbox"/>	Short	SMALLINT	<input checked="" type="checkbox"/>		3	0	
TimeStamp	TimeStamp	<input type="checkbox"/>	Date	DATETIME	<input checked="" type="checkbox"/>	"yyyyMMddhhmmss"	14	0	
ServiceKey	ServiceKey	<input type="checkbox"/>	Short	SMALLINT	<input checked="" type="checkbox"/>		1	0	
CallingPartyNum...	CallingPartyNumber	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		64	0	
APN	APN	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		22	0	
URL	URL	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		22	0	
CallingPartyIMSI	CallingPartyIMSI	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		64	0	
AccessNetworkA...	AccessNetworkAddr...	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		22	0	
GSNAddress	GSNAddress	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		22	0	
CallingRoamInfo	CallingRoamInfo	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		20	0	
CallingCellID	CallingCellID	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		20	0	
TimeStampOfSG...	TimeStampOfSGSN	<input type="checkbox"/>	Date	DATETIME	<input checked="" type="checkbox"/>	"yyyyMMddhhmmss"	14	0	
TimeZoneOfSGSN	TimeZoneOfSGSN	<input type="checkbox"/>	Short	SMALLINT	<input checked="" type="checkbox"/>		2	0	
BearerCapability	BearerCapability	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		32	0	
ChargingTime	ChargingTime	<input type="checkbox"/>	Date	DATETIME	<input checked="" type="checkbox"/>	"yyyyMMddhhmmss"	14	0	
TotalFlux	TotalFlux	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		50	0	
UpFlux	UpFlux	<input type="checkbox"/>	Long	BIGINT	<input checked="" type="checkbox"/>		8	0	
DownFlux	DownFlux	<input type="checkbox"/>	Long	BIGINT	<input checked="" type="checkbox"/>		8	0	
ElapsedTime	ElapsedTime	<input type="checkbox"/>	Integer	INT	<input checked="" type="checkbox"/>		4	0	
TerminationReas...	TerminationReason	<input type="checkbox"/>	Short	SMALLINT	<input checked="" type="checkbox"/>		2	0	
IMEI	IMEI	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		24	0	
TransitionID	TransitionID	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		32	0	
ServiceID	ServiceID	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		32	0	
SPID	SPID	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		32	0	
CategoryID	CategoryID	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		20	0	
ContentID	ContentID	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		20	0	
QoS	QoS	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		30	0	
OriginalNetwork...	OriginalNetworkType	<input type="checkbox"/>	Short	SMALLINT	<input checked="" type="checkbox"/>		20	0	
DiameterSessionID	DiameterSessionID	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		128	0	
BrandID	BrandID	<input type="checkbox"/>	Short	SMALLINT	<input checked="" type="checkbox"/>		3	0	
SubCOSID	SubCOSID	<input type="checkbox"/>	Integer	INT	<input checked="" type="checkbox"/>		7	0	
ChargingPartyNu...	ChargingPartyNumb...	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		64	0	
PayType	PayType	<input type="checkbox"/>	Short	SMALLINT	<input checked="" type="checkbox"/>		1	0	
BillCycleID	BillCycleID	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		20	0	

Figura 22 - Schema da tabela TB_DATA

Não foi possível apresentar todos os campos dessa tabela pois são cerca de 219 atributos, entre eles o File_Id que é uma chave estrangeira.

III. TB_FileError

Essa tabela possui o nome dos ficheiros que foram extraídos, mas que por alguma razão não foram inseridos com sucesso, ou ainda por *timeout* de FTP ou BD.

A figura a seguir mostra os campos para essa tabela.

Huawei (Output)									
Column	Db Colu...	Key	Type	DB Type	<input checked="" type="checkbox"/> N..	Date Pattern (...)	Len...	Prec...	I
filename	filename	<input type="checkbox"/>	String	VARCHAR	<input checked="" type="checkbox"/>		50		
date	date	<input type="checkbox"/>	Date	DATETIME	<input checked="" type="checkbox"/>	"dd-MM-yyyy"			

Figura 23 - Atributos da tabela TB_FileError

Essa tabela não altera com cada ficheiro, pois ela só precisa armazenar os ficheiros que houve algum erro no processamento.

4.3 O projeto desenvolvido no Talend

O desenvolvimento desse projeto baseia-se muito no processo ETL (utilizado principalmente para extracção e *load* de dados), onde foi utilizado a ferramenta Talend Open Studio.

Para execução desse projeto foi criado 7 Jobs que fará a extração dos dados, uma pequena transformação e depois a inserção dos ficheiros na tabela de controlo e os dados referentes a esse ficheiro na tabela de dados. Tudo isso usando um controlo que vamos criar.

Os Jobs criados servem como o modelo para outros ficheiros, o que pode alterar são os *schemas* e as tabelas. Vamos listar esses Jobs de acordo como são executados:

- Job dataFileError;
- Job dataConfigFile;

- Job dataListFTPLastNDir;
- Job dataListFileDir;
- Job dataExtractFile;
- Job dataLoadWithControl.

4.3.1 Job dataFileError

Esse Job foi criado para inserir na tabela TB_FileError, todos os ficheiros que por alguma razão não foi possível terminar o seu processamento.

Essa é o primeiro Job e é responsável por executar os demais Jobs que vamos falar. A figura a seguir mostra os fluxos desse Job.

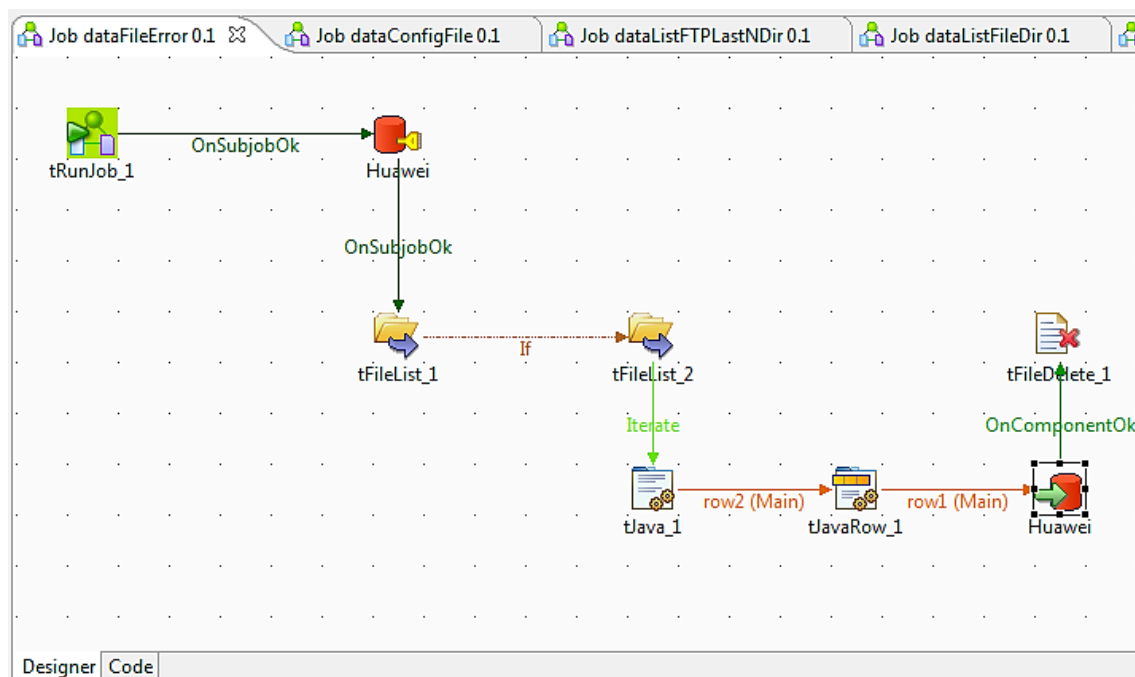


Figura 24 - Fluxo do Job dataFileError

Esse Job é responsável pela execução do próximo Job que será o Job dataConfigFile.

4.3.2 Job dataConfigFile

Esse Job é responsável pela leitura de um ficheiro de configuração, nesse ficheiro existe alguns dados que são importantes para o procedimento:

- **Directoria FTP** – a directoria principal onde se encontra as outras directorias (o nome dessa outra directoria varia de acordo com o dia Ex: no dia 24 de 06 de 20013 o nome dessa seria 20130624). Ex: como estamos a utilizar os ficheiros de CBS Data, a directoria FTP nesse caso poderia ser /cbs/data.
- **Ficheiro Texto** - o caminho de um ficheiro texto onde serão colocados as N nomes das directorias encontradas na Directoria FTP.
- **Um Número** – que corresponde a N nome de directorias que serão colocados no ficheiro texto.

A figura a seguir mostra o fluxo desse Job.

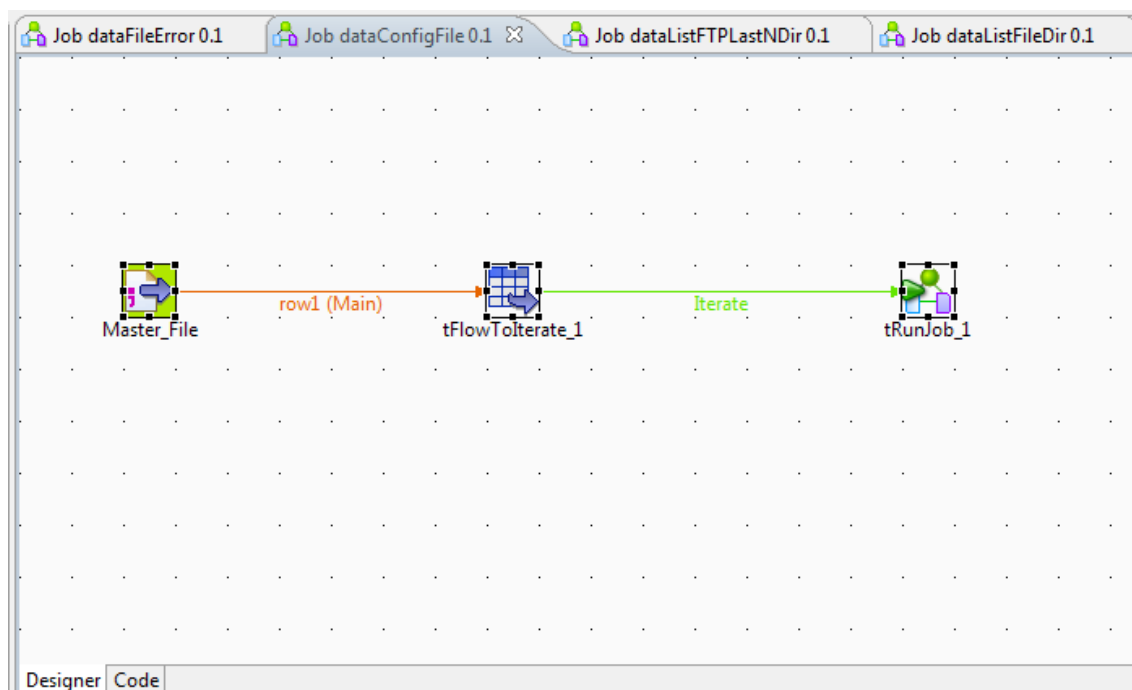


Figura 25 - Fluxo do Job dataConfigFile

Após a leitura do ficheiro esses dados serão enviados por parâmetros para o próximo Job, dando assim continuidade do processo.

4.3.3 Job dataListFTPLastNDir

Com os dados recebidos do Job anterior, esse vai abrir uma conexão com o FTP, onde vai na directoria principal e carrega os últimos N nomes das directorias que se encontram dentro da directoria principal e coloca num ficheiro texto de acordo com os dados recebidos.

A figura a seguir mostra o fluxo desse Job.

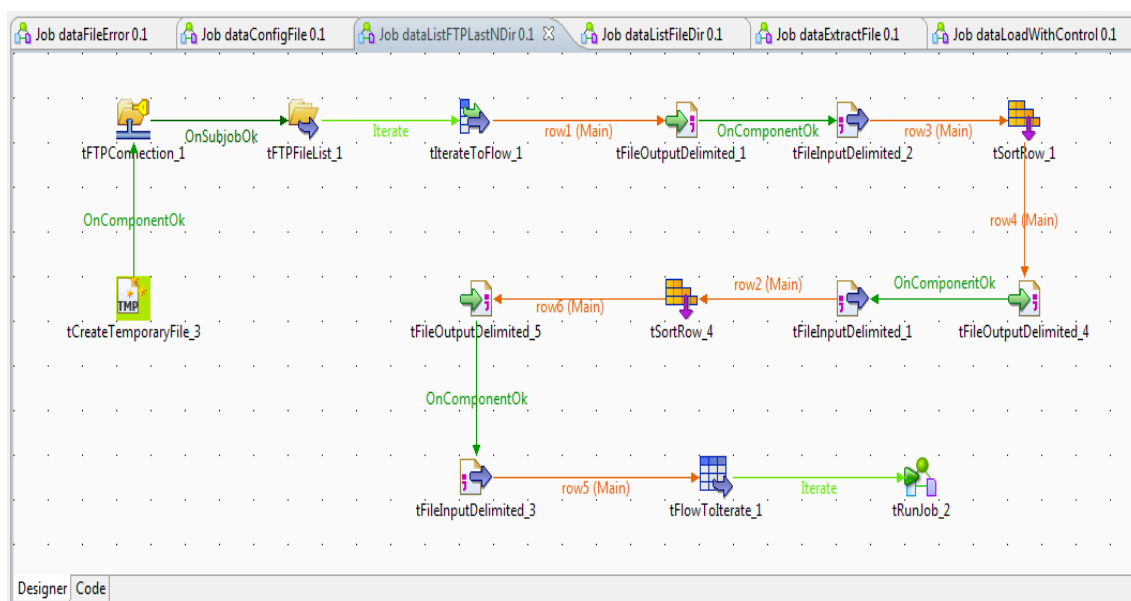


Figura 26 - Fluxo do Job dataListFTPLastNDir

Após a inserção de todos os N nomes das directorias no ficheiro texto, esse mesmo Job faz uma iteração, onde para cada nome de directoria, nesse ficheiro, será executado so próximo Job passando como parâmetro o nome da directoria.

4.3.4 Job dataListFileDir

Por cada nome de directoria que recebe do Job anterior, esse Job abre uma conexão com FTP, e vai na directoria e lista todos ficheiros que estão nessa directoria cujo nome começa com “data”. E com esses nomes listados vai a Base de Dados e lista todos os ficheiros cujo Flag é 2, ou seja, os ficheiros que já foram inseridos com sucesso.

Com as duas listas faz um cruzamento entre essas listas para saber quais os ficheiros que não foram inseridos e colocar num outro ficheiro texto.

A figura a seguir mostra o fluxo desse Job.

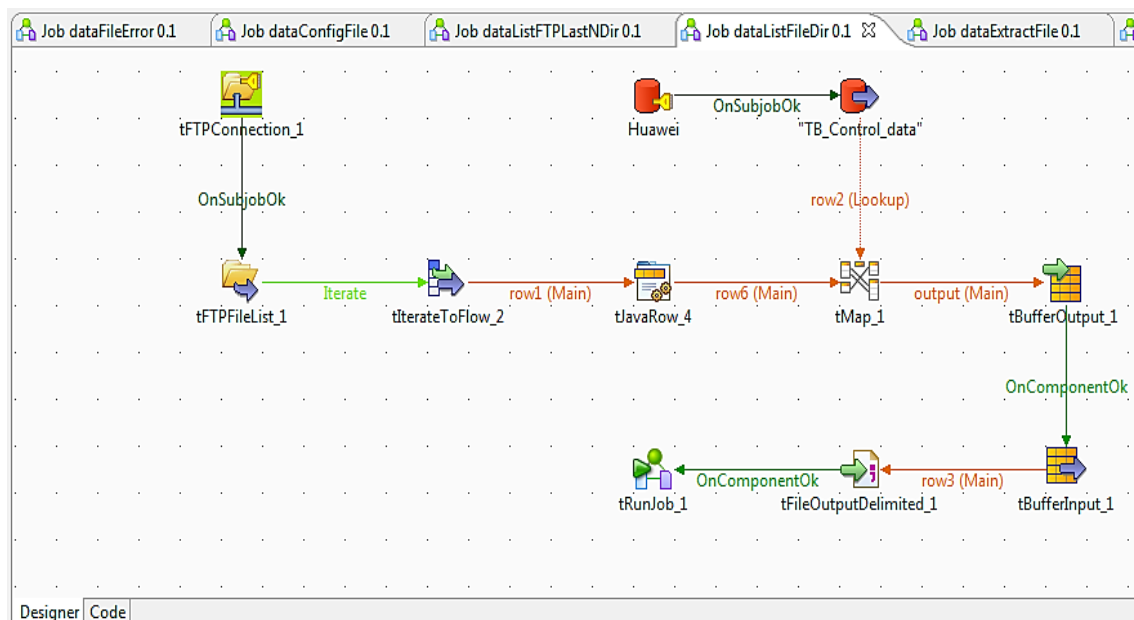


Figura 27 - Fluxo do Job dataListFileDir

Com a lista dos ficheiros que não foram inseridos, este Job executa o próximo Job, enviando para esse o caminho onde se encontra o ficheiro texto que possui a lista dos ficheiros não inseridos e a directoria FTP onde se encontram esses ficheiros.

4.3.5 Job dataExtractFile

Esse Job faz a leitura do ficheiro texto que contém os nomes dos ficheiros não inseridos e faz a extração de cada um desses ficheiros do FTP na directoria recebida do Job anterior.

A figura a seguir mostra o fluxo desse Job.

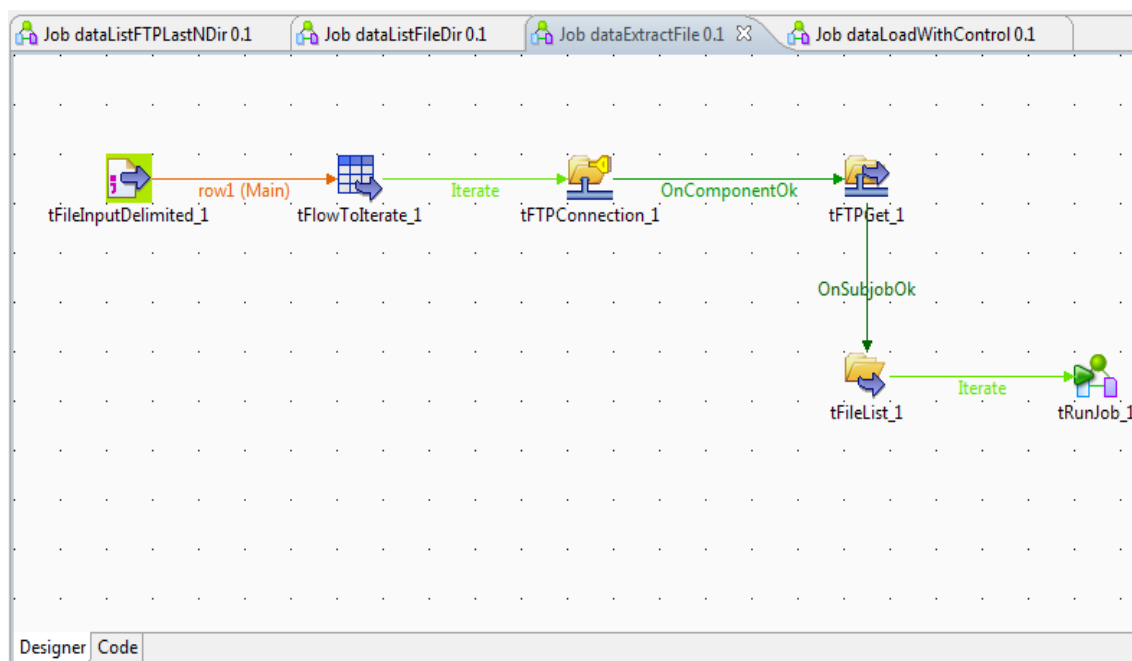


Figura 28 - Fluxo do Job dataExtractFile

Para cada ficheiro extraído este Job executa o próximo Job passando como parâmetro o nome desse ficheiro e o caminho onde se encontra esse ficheiro.

4.3.6 Job dataLoadWithControl

Ao receber o nome do ficheiro, este Job faz mais uma verificação para certificar que esse ficheiro não foi ainda inserido com sucesso na Base de Dados. E caso não estiver inserido, criamos uma conexão com a BD e fazemos a inserção desse ficheiro na tabela de controlo **TB_Control_DATA**, com as seguintes informações:

- **Id** (*bigint*) – incrementa 1 valor a partir da última linha inserido;
- **nameFile** (*varchar*) – nome do ficheiro recebido;
- **seq** (*varchar*) – a sequencia do ficheiro recebido;
- **flag** (*int*) – com valor 1, porque ainda não foi inseridos os dados desse ficheiro na tabela de dados, nesse caso a tabela **TB_DATA**;

- **date_modify** (*datetime*) – dia e hora em que está a ocorrer o processamento;
- **Total_File_Record** (*int*) – com valor zero (0), porque ainda não foi lido o ficheiro;
- **Total_insert_Record** (*int*) – com valor zero (0), porque ainda não foi inserido o ficheiro na tabela de dados **TB_DATA**.

Após a inserção desse ficheiro na tabela de controlo, vamos pegar o último Id inserido que corresponde ao ficheiro inserido para podermos utilizar no relacionamento entre a tabela de dados e a de controlo. O valor do Id será guardado num contexto para ser utilizado posteriormente na tabela de dados. Se tudo estiver ok fazemos o *commit* na tabela de controlo e fechamos a conexão.

Em seguida criamos uma nova conexão para inserirmos os dados desse ficheiro na tabela de dados.

Com a conexão aberta, fazemos a leitura do ficheiro que foi extraído, que foi enviado pelo Job anterior, e inserimos os dados desse ficheiro na tabela de dados, nesse caso **TB_DATA**, onde para cada linha inserido o File_Id será igual ao Id inserido na tabela de controlo, nesse caso será o valor que foi guardado no contexto.

A figura a seguir mostra o fluxo desse Job.

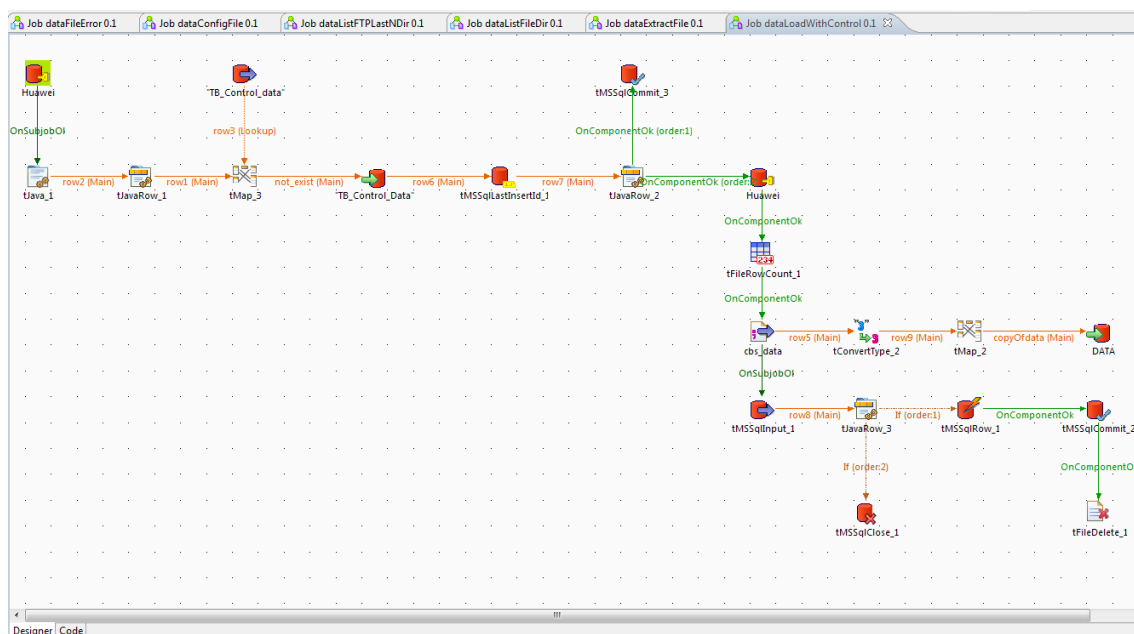


Figura 29 - Fluxo do Job dataLoadWithControl

Após a “inserção” dos dados do ficheiro na tabela de dados, vamos verificar se os dados inseridos na tabela referentes a esse ficheiro, são iguais aos que existem no ficheiro, ou seja, o número de linhas desse ficheiro tem que ser igual ao número de linhas inseridos na tabela **TB_DATA** com File_id igual ao Id da tabela de controlo que foi guardado no contexto. Dissemos “inserção”, pois ainda não foi feito o “commit” para a conexão que criamos.

Se forem iguais, temos que confirmar na tabela de controlo **TB_Control_DATA** que esse ficheiro foi inserido com sucesso, por isso vamos na tabela de controlo e atualizamos os seguintes campos quando Id igual ao Id guardado no contexto:

- **flag** (*int*) – com valor 2, para confirmar que todos os dados desse ficheiro já foram inseridos com sucesso na tabela de dados, nesse caso a tabela **TB_DATA**;
- **date_modify** (*datetime*) – dia e hora em que está a ocorrer o processamento;
- **Total_File_Record** (*int*) – número total de linhas que esse ficheiro possui;

- **Total_insert_Record** (*int*) – número total de linhas inserido na tabela de dados **TB_DATA**, que correspondem a esse ficheiro.

Quando a atualização for bem-sucedida vamos fazer o *commit* para manter esses dados inseridos na tabela de controlo e de dados. Depois terminamos a conexão e apagamos o ficheiro que recebemos do Job anterior.

Mas caso houver algum erro com o fluxo na figura anterior, nós não vamos validar os dados inseridos. Isso ajuda a garantir que não haverá nem duplicação e nem omissão de dados. Depois terminamos a conexão, mas nesse caso não vamos apagar o ficheiro, para que esse seja reportado na tabela **TB_FileError** como um ficheiro que não foi inserido com sucesso.

Após término desse processo o Job inicial dataFileError vai verificar todos os ficheiros que foram extraídos mas não foram inseridos nas tabelas e inseri-los na tabela **TB_FileError**.

4.4 A avaliação do Sistema

Com o término desse projeto, ele foi logo colocado na produção para que os novos ficheiros fossem inseridos usando o novo sistema de controlo. Desde então esse sistema funcionou muito bem.

Com o sistema criado é possível saber se um ficheiro foi ou não inserido, os dados duplicados desapareceram, e é impossível ter um ficheiro inserido em que os dados estejam a faltar na tabela de dados, ou seja é impossível ter dados omitidos para um ficheiro qualquer que esteja marcado com sucesso na tabela de controlo.

Uma das grandes vantagens para além de controlar os ficheiros (CDR's), foi que esse sistema serviu como um modelo para criar outros que vieram surgindo de acordo com as diversas necessidades.

4.5 Comparação entre o Antes e o Depois

Baseado na entrevista, e no acompanhamento do novo sistema e tendo em conta o sistema antigo, a tabela a seguir mostra o quadro comparativo entre o sistema que era utilizado antes e o sistema que foi criado depois.

	Sistema Antes	Sistema Depois
Tempo	Demorava muito tempo, um dia de dados poderia ser processados em Horas.	O tempo de Processamento melhorou muito, um dia de dados são processados em menos de 5 minutos.
Tabelas utilizadas	Era utilizado duas tabelas, uma para colocar o nome do ficheiro e outra para colocar os dados desse ficheiro. Só que não existia relação entre essas tabelas	Foi criado também duas tabelas, uma para colocar o nome do ficheiro e outra para colocar os dados desse ficheiro. Mas com relações entre as duas tabelas indicando a que ficheiro pertence cada registo na tabela de dados.
Tinha controlo?	Não tinha controlo.	Tem um forte controlo na extração e inserção dos ficheiros e os dados referentes a esse ficheiro.
Relação ficheiros versus dados	Como não existia uma relação entre a tabela de ficheiros e a de dados era impossível saber a que ficheiro pertencia um determinado registo na tabela de dados.	Com a relação entre duas tabelas é possível ver os ficheiros e os dados relacionados a esses ficheiros na tabela de dados.
Possui dados duplicados?	É possível encontrar dados duplicados, porque não havia um controlo dos ficheiros.	Com o controlo criado é “impossível” ter dados duplicados.
É possível ter dados omitidos?	Por não haver um controlo, é possível haver falta de dados na tabela	Não. O controlo garante que se um ficheiro for inserido todas as linhas pertencentes a esse ficheiro são inseridos na tabela de dados. E caso contrário esse não será inserido.
Possui dados nulos?	Também era possível existir dados nulos, pois não havia um tratamento dos dados antes de inseridos.	Os dados são examinados e inseridos.

Tabela 7 – Comparação entre o sistema antigo e o atual.

5 Considerações finais

Apesar do esforço inicial para entender um novo *software* (ferramenta), esse projeto foi muito vantajoso, tendo em conta que esse foi o primeiro contacto com sistemas ETL. Tendo em conta o sistema antigo, e os problemas que existiam, podemos constatar que o sistema criado trouxe mais capacidade de controlo dos ficheiros que são inseridos na BD. E com um maior controlo, diminui-se o trabalho que era feito manualmente para tentar encontrar os possíveis erros ou falhas que podia haver com o sistema, que mesmo assim ainda nem sempre todas as falhas eram encontradas.

Com o novo sistema transaccional criado não é possível a duplicação dos dados, nem a sua omissão, garantindo assim a integridade dos dados que se encontram na base de dados. Também ele permite saber a que ficheiro pertence cada registo na tabela de dados, por outro lado, permite saber quantos registos na tabela de dados pertence a um determinado ficheiro na tabela de controlo.

Uma das maiores vantagens desse sistema é que ele serviu como um modelo para quase todos os ficheiros que são processados diariamente e armazenados numa base de dados.

Conclusão

O processo ETL surgiu para dar suporte a as organizações no que tange a integração de dados, permitindo que essas organizações tenham dados de todas as fontes bem transformados, conforme as suas necessidades. Esses dados podem estar em um ou vários repositórios finais.

No desenrolar deste trabalho vimos que o processo ETL, é atualmente um factor importante para as empresas, principalmente para tratamentos dos dados. Também destacamos as principais etapas desse processo como sendo a **Extração**, **Transformação** e **Load** de dados, em que:

- A **Extração** é a fase onde os dados são extraídos de um ou mais fontes de dados.
- **Transformação** é a segunda fase onde os dados extraídos passam por uma limpeza e transformação, para ficarem em conformidade com as necessidades dos utilizadores finais. Pode-se ver que há dados em que não é necessário transformações. Essa etapa é a que consome mais recurso e tempo.
- **Load** é a última fase em que os dados transformados (ou extraídos em transformação), são carregados (inseridos) para um ou vários repositórios finais.

De entre essas etapas a extração e o *Load* são as obrigatórias, sendo a etapa de transformação opcional.

Constatamos ainda que esse processo não é só “Extração”, “Transformação” e “Load”, dado que existem subprocessos que auxiliam essas três. Portanto, as três (3) etapas supracitadas são as clássicas, que podem ser expandido para quatros (4), **extração, limpeza, confirmação, e entrega.**

Num projeto de DW esse processo consome quase 70% dos recursos e tempos disponibilizados. Por isso é considerado como o processo mais crítico na construção de um DW.

Quanto as ferramentas de ETL, apresentamos algumas soluções que existe no mercado para quem precisa desenvolver seu sistema de ETL. Mas também é possível criar uma ferramenta manualmente, apesar de responder exatamente as necessidades pretendidas, não é muito aconselhado, dado que, é cansativo, exige profissionais experientes e inteligentes com habilidades e especialidades.

Em relação ao caso prático, ficou claro que é possível ter um sistema ETL que não seja para um DW. Constatou-se ainda que TOSDI é a ferramenta de ETL mais utilizada na empresa Unitel T+, porque fornece uma interface mais intuitiva, é *OpenSource*, e possui uma grande variedade de componentes que facilitam no desenvolvimento do *workflow*. Também TOSDI possui um grande número de conectores para diferentes Base de Dados.

O sistema possuía alguns problemas como:

- Falta de controlo dos ficheiros, ou seja era difícil saber quais ficheiros foram inseridos e quais ainda faltavam por inserir.
- Era difícil saber a que ficheiro pertence um determinado registo na tabela de dados.

- Duplicação e omissão dos dados.
- Grandes aumentos do tráfego na rede porque o sistema não conseguia saber quais ficheiros já tinham sido carregados.
- O tempo de processamento era elevado comparado com o sistema atual.
- Possuía muitos valores nulos, porque não havia uma transformação eficiente dos dados que são carregados para BD.

A proposta e a solução apresentada para resolução desses problemas foi o seguinte:

- Criação de duas tabelas relacionais, em que a primeira tabela armazena as informações a cerca do ficheiro e a segunda a cerca dos dados que correspondem a esse ficheiro.
- Criação de um sistema ETL que tenha controlo na extração e na inserção de dados, em que:
 - O **controlo na extração** não permite que um ficheiro que já tenha sido inserido com sucesso seja novamente extraído. Eliminando a sobrecarga na rede.
 - O **controlo na inserção** não permite que tenhamos dados duplicados ou omitidos, porque usa um “flag” (2 se com sucesso e 1 caso contrário). O “*flag*” é igual a 2, se todas as linhas do ficheiro foram inseridas na tabela de dados. E caso houver alguma linha que não foi inserida, logo esses dados não serão inseridos, e o “*flag*” é igual a 1, para indicar que esse ficheiro não foi inserido com sucesso, sendo assim na próxima extração esse ficheiro será novamente extraído.

Esse projeto exigiu muita dedicação, pois foi utilizado uma tecnologia que era nova, por isso o principal desafio foi aprender a usar a ferramenta TOSDI. Também foi um desafio a compreensão do sistema antigo para que pudesse criar uma solução que fosse viável.

Portanto, foi um grande desafio mas o resultado foi satisfatório para ambas as partes, porque ganhou-se mais conhecimento, e a empresa Unitel T+ ganhou no processamento dos dados, que por sua vez permite ter relatórios de qualidades e a tomada de decisão seja a melhor possível.

Também esse projeto serviu como um modelo para criação de novos que vieram surgindo. Esse modelo foi utilizado para processamento de todos os ficheiros de MSC e CBS.

O sistema ETL desenvolvido para a empresa Unitel T+ foi importante para o processamento dos dados, porque esses dados são inseridos na BD usando um forte controlo entre os ficheiros e os dados referentes a esses ficheiros. E com a nova metodologia introduzida, o trabalho da equipa de IT ficou mais fácil, visto que não precisam estar todos os dias a apagar e inserir os dados.

Em conclusão, o processo ETL é crucial para tratamento dos dados, facilitando a integração entre diferentes fontes, e permitindo que esses dados sejam entregues para um ou vários repositórios finais, de acordo com as necessidades da organização. Em suma, esse processo tenta satisfazer as necessidades dos utilizadores finais.

Bibliografia

- Altos, L. (22 de 01 de 2008). *Talend to Optimize Open Source Data Integration on Windows Platform*. Obtido em 18 de 06 de 2013, de Talend:
<http://www.talend.com/about-us/press/talend-optimize-open-source-data-integration-windows-platform>
- Becker, B. (21 de 10 de 2007). *Subsystems of ETL Revisited*. Obtido em 22 de 04 de 2013, de Kimball Group:
<http://www.kimballgroup.com/2007/10/21/subsystems-of-etl-revisited/>
- Bowen, J. (2012). *Getting Started with Talend Open Studio for Data Integration*. Birmingham: Packt Publishing.
- Brodkin, J. (24 de 04 de 2007). *Open source start-up upgrades data integration software*. Obtido em 18 de 06 de 2013, de NetworkWorld:
<http://www.networkworld.com/news/2007/042407-talend-data-integration.html?page=1>
- Chapple, M. (sd). *Flat File*. Obtido em 26 de 06 de 2013, de About.Com:
<http://databases.about.com/cs/administration/g/flatfile.htm>
- Dwh. (25 de 04 de 2011). *Buy vs. Build*. Obtido em 10 de 06 de 2013, de ETL TOOLS:
<http://dwhtltool.blogspot.com/2011/04/buy-vs-build.html>
- Ferreira, J., Miranda, M., Abelha, A., & Machado, J. (10 de 09 de 2010). *ACTAS/PROCEEDINGS*. Obtido em 20 de 11 de 2012, de INForum:
<http://inforum.org.pt/INForum2010/papers/sistemas-inteligentes/Paper080.pdf>
- Ferreira, R. (3 de 07 de 2007). *Implementando SOA nas Empresas*. Obtido em 20 de 04 de 2013, de Ricardo Ferreira's Blog: <http://architecture-journal.blogspot.com/2007/07/implementando-soa-nas-empresas.html>
- Gama, F. S., & Abreu. (12 de 2008). *Desmistificando o Conceito de Etl*. Obtido em 26 de 04 de 2013, de FSMA:
http://www.fsma.edu.br/si/Artigos/V2_Artigo1.pdf

- Hammergren, T. C., & Simon, A. R. (2009). *Data Warehousing For Dummies* (2 ed.). Indianapolis, IN: Wiley Publishing, Inc.
- Hoffer, J. A., Ramesh, V., & Topi, H. (2010). *MODERN DATABASE MANAGEMENT*. New Jersey: Pearson.
- Inmon, B. (08 de 05 de 2008). *A Brief History of ETL*. Obtido em 16 de 12 de 2012, de Channel: BI & Integration - Colin White: <http://www.b-eye-network.com/channels/1138/view/7040>
- Jiang, B. (02 de 05 de 2013). *Data Warehouse Construction: Forces Driving Architectural Evolution*. Obtido em 30 de 05 de 2013, de BeyeNETWORK: <http://www.b-eye-network.com/view/16930>
- Josko, J. M. (s.d.). *Uma análise das alternativas de arquiteturas para a integração de dados*. Obtido em 20 de 04 de 2013, de DevMedia: <http://www.devmedia.com.br/uma-analise-das-alternativas-de-arquiteturas-para-a-integracao-de-dados/9240>
- K, K. (28 de 06 de 2012). *Difference between Full load & Incremental load*. Obtido em 09 de 04 de 2013, de DATAWAREHOUSE CONCEPTS: <http://dwhlaureate.blogspot.com/2012/06/what-is-full-load-incremental-load-full.html>
- K, K. (28 de 06 de 2012). *What does ETL tool mean?* Obtido em 20 de 05 de 2013, de DATAWAREHOUSE CONCEPTS: <http://dwhlaureate.blogspot.com/2012/06/what-is-anetl-tool-mean-etl-tools-are.html>
- K, K. (03 de 03 de 2013). *What is Online and Offline Extraction?* Obtido em 04 de 04 de 2013, de Datawarehouse Concepts: <http://dwhlaureate.blogspot.com/2013/03/what-is-online-and-offline-extraction.html>
- Kakish, K., & Kraft, T. A. (2012). *ETL Evolution for Real-Time Data Warehousing. Conisar Proceedings* (p. 12). New Orleans Louisiana, USA: EDSIG (Education Special Interest Group of the AITP) .

- Kimball, R., & Caserta, J. (2004). *The data warehouse ETL toolkit : practical techniques for extracting, cleaning, conforming, and*. Indianapolis: Wiley Publishing, Inc.
- Kimball, R., & Ross, M. (2010). *The Kimball Group Reader: Relentlessly Practical Tools for Data Warehousing and Business Intelligence*. Indianapolis: Wiley Publishing, Inc.
- Kimball, R., Ross, M., Thornthwaite, W., Mundy, J., & Becker , B. (2008). *The Data Warehouse Lifecycle Toolkit, Second Edition*. Indianapolis: Wiley Publishing.
- Kozielski, S., & Wrembel, R. (2008). *New Trends in Data Warehousing and Data Analysis* . New York: Springer Science+Business Media.
- Lane, P. (2005). *Oracle Database Data Warehousing Guide*. Redwood : Oracle.
- Lima, C. A. (14 de 04 de 2010). *Os Subsistemas de ETL – 34 Subsistemas Revisados*. Obtido em 23 de 04 de 2013, de Blog do Lito – Data Warehouse / Business Intelligence: <http://litolima.com/2010/04/14/os-subsistemas-de-etl-34-subsistemas-revisados/>
- Liu, L., & Özsu, M. T. (2009). *Encyclopedia of Database Systems*. New York: Springer.
- Lynch, W. (13 de 03 de 2012). *The Best Open-Source ETL Tools*. Obtido em 09 de 06 de 2013, de Ask: <http://www.ask.com/explore/opensource-etl-tools-3962>
- Madsen, M. (01 de 10 de 2004). *Deciding to Buy or Build ETL for Your Data Warehouse*. Obtido em 10 de 06 de 2013, de information management: <http://www.information-management.com/issues/20041001/1011015-1.html>
- Misra, H., & Rahman, H. (2013). *Managing Enterprise Information Technology Acquisitions*. Hershey: IGI Global.
- Montcheil, Y. d., & Dupupet, C. (07 de 03 de 2006). *Third Generation ETL: Delivering the Best Performance*. Obtido em 19 de 05 de 2013, de BeyeNETWORK: <http://www.b-eye-network.com/view/2511>

- Mundy, J. (06 de 04 de 2008). *Should You Use An ETL Tool?* Obtido em 12 de 06 de 2013, de Kimball Group: <http://www.kimballgroup.com/2008/04/06/should-you-use-an-etl-tool/print/>
- Mussi, C. (2004). *Data Warehouse - Da Modelagem a Implantação*. p. 16.
- Neto, T. C. (07 de 05 de 2012). *Pentaho - Business Intelligence*. Obtido em 08 de 08 de 2012, de Ambiente Livre: http://www.ambientelivre.com.br/downloads/doc_download/87-tcc-ferramentas-de-etl-open-source-talend-e-kettle.html
- Niso. (2004). *Understanding Metadata*. Obtido em 13 de 05 de 2013, de Niso: <http://www.niso.org/standards/resources/UnderstandingMetadata.pdf>
- Ponniah, P. (2010). *Data warehousing fundamentals for IT professionals*. New Jersey: John Wiley & Sons, Inc.
- Ribeiro, V. (28 de 06 de 2011). *O que é ETL?* Obtido em 06 de 04 de 2013, de Viviane Ribeiro: <http://vivianeribeiro1.wordpress.com/2011/06/28/o-que-e-etl-2/>
- Ross, M., & Kimball, R. (13 de 11 de 2004). *Surrounding the ETL Requirements*. Obtido em 08 de 04 de 2013, de Kimball Group: <http://www.kimballgroup.com/2004/11/10/surrounding-the-etl-requirements/>
- Searls, D. B. (2003). *Data integration—connecting the dots*. Obtido em 17 de 04 de 2013, de Nature Biotechnology: <http://www.nature.com/nbt/journal/v21/n8/full/nbt0803-844.html>
- Sherman, R. (19 de 02 de 2009). *Beyond ETL and Data Warehousing*. Obtido em 09 de 04 de 2013, de Athena IT Solutions: http://www.information-management.com/infodirect/2009_109/-10014988-1.html?zkPrintable=1&nopagination=1
- Souza, M. d. (28 de 11 de 2003). *Metadados*. Obtido em 13 de 05 de 2013, de iMasters: <http://imasters.com.br/artigo/1569/gerencia-de-ti/metadados/>

- Tableau. (2012). *Refreshing Extracts*. Obtido em 04 de 04 de 2013, de Tableau:
http://onlinehelp.tableausoftware.com/current/pro/online/en-us/extracting_refresh.html
- Talend. (sd). *Talend Open Studio : User Guide*. Talend.
- Thoo, E., Friedman, T., & Beyer, M. (18 de 10 de 2012). *Magic Quadrant for Data Integration Tools*. Obtido em 20 de 03 de 2013, de Gartner:
<http://www.gartner.com/technology/reprints.do?id=1-1CYG9N1&ct=121127&st=sb>
- Tools, E. (2011). *Extract Transform Load*. Obtido em 06 de 04 de 2013, de ETL Tools:
<http://www.etltools.org/>
- Vaz, M. S. (sd). *Introdução a Metadados*. Obtido em 13 de 05 de 2013, de DevMedia:
<http://www.devmedia.com.br/introducao-a-metadados/1883>
- Wang, J. (2008). *Encyclopedia of Data Warehousing and Mining Second Edition*. Hershey: Information Science Reference.
- Yuhanna, N. (2012). *The Forrester Wave™: Enterprise ETL, Q1 2012*. Forrester .
- Zode, M. (24 de 10 de 2011). *The Evolution of ETL*. Obtido em 19 de 05 de 2013, de docshut: http://www.scribd.com/fullscreen/70126065?access_key=key-2gv0i58hkoec0tdm4pq2

A Guião de entrevista

O presente guião destina-se aos três entrevistados o Elton Correia, Denize Oliveira e Inácio Carvalho, e tem como objetivo saber do uso do sistema ETL na empresa Unitel T+ Telecomunicações, para auxiliar na análise do caso prático.

1. Possuem um sistema ETL?
2. Qual foi o 1º método utilizado?
3. Em que consistia a DTT?
4. Como era feito o controlo dos ficheiros que são carregados para Base de Dados?
5. Qual era a duração do processamento?
6. Utilizam alguma ferramenta ETL?
7. O levou a optar por ferramentas ETL e qual foi a 1ª adotada e em que versão?
8. Por que deixaram de utilizar o Jasper ETL e optar por Talend Open Studio?
9. Pretendem utilizar outra ferramenta?

B Os entrevistados

Nome	Cargo	Contato
Elton Correia	Chefe Desenvolvimento e Reporting	elton.correia@tmais.cv
Denize Oliveira	Engenheira de Reporting	denize.oliveira@tmais.cv
Inácio Carvalho	<i>IT Manager</i>	Inacio.carvalho@tmais.cv