u·évora
UNIVERSIDADE DE ÉVORA

DEPARTAMENTO DE INFORMÁTICA

Mestrado em Engenharia Informática

Dissertação

Metodologias de Qualidade e Testes de Data Warehouses - sua importância para a flexibilização e optimização do processo de Data Warehouse

"Esta dissertação não inclui as críticas e sugestões feitas pelo júri"

Francisco Luís Marinho do Rosário Matias (nº 17386)

ORIENTADOR: PAULO QUARESMA
COORDENADOR EXTERNO: JOÃO DUARTE

u·évora
UNIVERSIDADE DE ÉVORA

DEPARTAMENTO DE INFORMÁTICA

Mestrado em Engenharia Informática

Dissertação

Metodologias de Qualidade e Testes de Data Warehouses - sua importância para a flexibilização e optimização do processo de Data Warehouse

"Esta dissertação não inclui as críticas e sugestões feitas pelo júri"

Francisco Luís Marinho do Rosário Matias (nº 17386)

ORIENTADOR: PAULO QUARESMA
COORDENADOR EXTERNO: JOÃO DUARTE

Agradecimentos

Em primeiro lugar gostaria de agradecer a toda a minha família e amigos, por todo o seu apoio e incentivos, especialmente aos meus pais que sempre me ajudaram. Sem eles com certeza que esta oportunidade não poderia ter sido concretizada.

Gostaria também de agradecer ao professor Paulo Quaresma da Universidade de Évora, pela orientação e ajuda ao longo deste projecto, bem como todos os ensinamentos que me transmitiu ao longo destes anos.

Um agradecimento muito especial para o Dr. João Duarte, pela forma como me recebeu e me ajudou incondicionalmente, mas também por me ter dado a oportunidade de puder realizar este projecto.

Gostaria também de expressar a minha gratidão a todos os meus colegas de trabalho, em especial ao Eng. Hugo Malarranha e ao Eng. Miguel Macedo.

Ao meu colega e amigo Sérgio Faustino, agradeço toda a sua ajuda e apoio nos momentos em que mais necessitei.

Por fim, mas não menos importante, a todos os meus colegas e professores que me acompanharam no meu percurso escolar e académico, um muito obrigado também.

Metodologias de Qualidade e Testes de Data Warehouses – sua importância para a flexibilização e optimização do processo de Data Warehouse

Conseguir informações que permitam a tomada de decisões baseadas num volumoso histórico de dados torna-se difícil quando as grandes empresas detêm um enorme volume de dados espalhados em diversos sistemas. Os sistemas de *Data Warehouse* surgiram da necessidade de transformar e organizar estes dados para a geração de informações estratégicas úteis. Com o avanço exponencial do progresso tecnológico, o ciclo de vida de desenvolvimento dos produtos torna-se menor, obrigando as empresas a tomarem decisões operacionais e estratégicas em curtos intervalos de tempo. Assim, a competitividade de uma organização é determinada pela sua capacidade de tomar decisões precisas que garantam o sucesso da sua estratégia de negócio. Essas decisões devem ser baseadas em informações actualizadas, completas e de elevada qualidade. A aplicação de metodologias de qualidade e testes assumem-se como essenciais para a manutenção e optimização do processo de *Data Warehouse*. Investir nestas metodologias é investir na qualidade da informação.

Data Warehouse quality methodologies and tests - their importance for the flexibility and optimization of the Data Warehouse process

Getting information that allow decision making based on a voluminous historical data becomes difficult when large companies have an enormous amount of data scattered in different systems. The Data Warehouse systems arose from the need to transform and organize the data for generating useful strategic information. With the exponential growth of the technological progress, the life cycle of product development becomes smaller, forcing companies to take strategic and operational decisions in short intervals of time. Thus, the competitiveness of an organization is determined by its ability to make accurate decisions to ensure the success of its business. These decisions should be based on updated, fulfilled and high quality information. The application of quality methodologies and tests take on as essential to the maintenance and optimization of the Data Warehouse process. Investing in these methods is to invest in the quality of information.

Índice Geral

Índice de Figuras

Índice de Tabelas

Termos abreviados, Definições

Termo	Descrição
BI	*Business Intelligence*
BICC	*Business Intelligence Competency Center*
BD	*Base de dados*
DBA	*Database Administrator*
DM	*Data Mart*
DSS	*Decision Support System*
DW	*Data Warehouse*
ECTS	*European Credit Transfer and Accumulation System*
EIS	*Executive Information System*
ETL	*Extract, Transform and Load*
FS	*Filesystem*
FTP	*File Transfer Protocol*
HOLAP	*Hybrid On Line Analitical Processing*
IDW	*Integration, Development and Warehousing*
LDAP	*Lightweight Directory Access Protocol*
MIT	*Massachusetts Institute of Technology*
MOLAP	*Multidimensional On Line Analitical Processing*
ODS	*Operacional Data Storage*
OLAP	*On-Line Analytical Processing*
OLTP	*On-Line Transaction Processing*
ROLAP	*Relational On Line Analitical Processing*
RTDW	*Real Time Data Warehouse*
SSL	*Secure Sockets Layer*
TI	*Tecnologias de Informação*

1 Introdução

1.1 Contexto Académico

Este documento foi escrito no contexto da dissertação final de curso do Mestrado em Engenharia Informática da Universidade de Évora. Trata-se de uma unidade curricular de 48 ECTS (*European Credit Transfer and Accumulation System*).

Como base da realização desta dissertação foi efectuado um estágio curricular, sendo a carga horária semanal média de 30 horas.

O estágio decorreu durante o período de 6 de Fevereiro de 2008 a 5 de Agosto de 2008.

A orientação pedagógica deste projecto esteve a cargo do Professor Paulo Quaresma do Departamento de Informática da Universidade de Évora.

O estágio curricular decorreu para a instituição IDW (*Integration, Development and Warehousing*) em *outsourcing* numa empresa de telecomunicações. A coordenação externa esteve a cargo do Dr. João Duarte.

1.2 Contexto Institucional

A instituição que me recebeu e possibilitou a realização deste projecto foi a IDW. Fundada em 2003, a IDW é uma empresa Portuguesa com apostas claras e inequívocas. Uma arquitectura de serviços e produtos coerente e consistente, que privilegia essencialmente a integração, as capacidades técnicas dos produtos e a capacidade de suporte e manutenção que pode ser oferecida. As principais áreas de actuação da empresa são: *Business Intelligence*, Segurança, Consultoria, *Business Process Automation*, *Data Management & Business Continuity* (IDW, 2003).

1.3 Contexto Tecnológico e Científico

Conseguir informações que permitissem a tomada de decisões baseadas num volumoso histórico de dados tornava-se numa missão difícil sendo necessário muitos recursos. Isto porque geralmente as grandes empresas detêm um

enorme volume de dados espalhados em diversos sistemas. Os sistemas de *Data Warehouse* surgiram da necessidade de conseguir dar um "rumo" a esta informação, permitindo assim avaliações de negócio ao mais alto nível. Essas avaliações devem ser baseadas em informações actualizadas, completas e de elevada qualidade. Nas últimas décadas a qualidade de *software* tem vindo a ser uma área cada vez mais importante e com a necessidade de melhorar cada vez mais a qualidade, têm vindo a crescer as metodologias de qualidade e testes.

1.4 Objectivos da Dissertação

Esta dissertação tem por objectivo em primeiro lugar, conceituar e enquadrar historicamente os sistemas de *Data Warehouse* bem como abordar as suas características, arquitecturas e modelos de arquitectura de dados. Pretende também mostrar a importância destes sistemas para uma empresa de telecomunicações, bem como todo o processo de construção e manutenção evolutiva envolvido. É também objectivo desta dissertação, demonstrar as metodologias de qualidade e testes que podem ser aplicadas aos projectos de *Data Warehouse*, mostrando como garantem a qualidade da informação a disponibilizar nestes sistemas. O estágio curricular foi a base para a descrição destas metodologias, permitindo que fosse feita por fim uma avaliação/apreciação crítica.

1.5 Estrutura do Documento

A estrutura desta dissertação está organizada pelos seguintes capítulos:

Data Warehouse – Este capítulo pretende enquadrar historicamente os sistemas de *Data Warehouse* e fornecer definições para estes sistemas de acordo com alguns autores. Pretende também descrever as características de um *Data Warehouse*, arquitecturas e modelos de arquitectura de dados, bem como os seus processos básicos.

Conceitos Envolvidos – Este capítulo descreve importantes conceitos envolvidos com o tema dos sistemas de *Data Warehouse*, tais como, *Business Intelligence, Executive Information Systems, Decision Support Systems, Business Intelligence Competency Center, Data Mart, Staging Area*, etc.

Importância de um *Data Warehouse* para uma organização de telecomunicações num ambiente em constante mutação – Tal como o título indica, este capítulo pretende mostrar a importância de um sistema de *Data Warehouse* para uma empresa de telecomunicações, tendo em conta o mercado altamente competitivo em que vivemos e de constante mudança.

O processo de construção e manutenção evolutiva de um *Data Warehouse* – Neste capitulo é descrito de um modo geral, o processo de contínuo desenvolvimento evolutivo e de eficiente manutenção que é necessário aos sistemas de *Data Warehouse.*

Abordagens e Metodologias de Qualidade e Testes de um Data *Warehouse* – Aqui são apresentadas abordagens e metodologias de qualidade e testes, tais como, o enquadramento de uma equipa de testes, operacionalização de uma equipa de testes, objectivos da aplicação destas metodologias, ciclo de desenvolvimento, tipos de testes, exemplos de testes, etc.

Caso Prático – Este capítulo, pretende demonstrar em termos práticos a aplicação de metodologias de qualidade e testes por parte de uma equipa de testes numa empresa de telecomunicações.

Conclusões – Neste capítulo é feita uma apreciação crítica ao trabalho desenvolvido, são apresentados resumidamente os principais projectos desenvolvidos no âmbito do estágio, e por fim são apresentadas as conclusões.

2 Data Warehouse

2.1 Enquadramento Histórico e Motivação

À medida que o progresso tecnológico avança a uma velocidade exponencial, o ciclo de vida de desenvolvimento dos produtos torna-se cada vez menor, obrigando as empresas a tomarem decisões operacionais e estratégicas em intervalos decrescentes de tempo. Dada esta situação, a competitividade de uma organização passa a ser determinada, em grande parte, pela sua capacidade de tomar decisões precisas que garantam o sucesso da sua estratégia de negócio. Essas decisões devem ser baseadas em informações actualizadas, completas e de elevada qualidade.

No início da década de 90, os sistemas *Data Warehouse* (DW) surgiram como uma solução para os problemas de apoio à decisão. A informação tornou-se o bem mais valioso dentro das instituições, e o mais interessante é que esta informação está disponível nas próprias organizações. Os dados para a obtenção de informações estão armazenados nas bases de dados operacionais, sistemas legados, arquivos pessoais e documentos. O grande desafio é saber como tornar estes dados úteis para o negócio, transformando dados em informações, e estas em acções que melhorem a gestão dos negócios. Para aceder a uma base de dados gigantesca e dispersa, e conseguir concretizá-la numa fonte de informação viável, é necessário transformar e organizar os dados para a geração de informações estratégicas úteis.

Como as decisões se fazem consoante a informação que está disponível e são tanto mais eficazes quanto melhor for essa informação, torna-se necessário reestruturar e repensar os sistemas de uma empresa com o objectivo de melhor disponibilizar a informação relevante para cada tipo de decisão. O tipo de informação que circula numa organização é específico para cada nível da estrutura organizacional. Assim, enquanto um gestor operacional necessita de informação para tratar e armazenar transacções diárias e necessárias ao negócio (suportado por sistemas transaccionais), a nível estratégico, a informação deverá ser proveniente de diversas fontes (internas e/ou externas), sob um ponto de vista analítico e centrada nos negócios (suportado por um sistema de apoio à decisão).

A evolução da tecnologia e o crescimento do uso de computadores ligados em rede, permitiu que médias e grandes empresas utilizassem sistemas informáticos para realizar os seus processos mais importantes, o que com o passar do tempo gerou uma quantidade enorme de dados relacionados com os negócios, mas que não estavam relacionados entre si. Esses dados armazenados num ou mais sistemas operacionais de uma empresa são um recurso, mas de modo geral, raramente servem como recurso estratégico quando se encontram ainda nesse estado original. Os sistemas informáticos convencionais não são desenvolvidos para gerar e armazenar as informações estratégicas, o que torna os dados vagos e sem valor para o apoio ao processo de tomada de decisão das organizações. Estas decisões são normalmente tomadas com base na experiência dos administradores da empresa, mas poderiam também ser baseadas em factos históricos armazenados pelos diversos sistemas de informação utilizados pelas organizações. Mediante esta situação surgiu o *Data Warehouse*, como um desafio para integrar esses dados, eliminando as redundâncias e identificando informações iguais que pudessem estar representadas sob formatos diferentes em sistemas distintos.

A importância dos dados como fonte não apenas do controlo operacional, mas da estratégia operacional tem sido a chave principal no conceito de *Data Warehouse*. Os dados são assim um elemento estratégico, sendo muitas vezes desperdiçados em suportes digitais que só existem devido a normas da empresa de registar os dados transaccionais por um determinado período de tempo. O *Data Warehouse* tornou-se teoricamente uma fonte de vantagem competitiva, no entanto, manter a competição já não é garantia de sobrevivência, é necessário surpreender a concorrência. Este sistema tornou-se numa peça muito útil nesta nova filosofia de negócio, utilizando hoje em dia um processo de armazenamento de dados que tem vindo a ganhar força devido à evolução do *hardware* e *software*. O objectivo do *Data Warehousing* é portanto fornecer a informação certa, para a pessoa certa, no momento exacto.

2.2 DEFINIÇÕES DE DATA WAREHOUSE

O professor Ralph Kimball é um dos maiores conceituados precursores dos conceitos de *Data Warehouse*. Define *Data Warehouse* como toda a fonte de informação "questionável" de uma empresa. Este autor refere que um DW não é mais do que a união de um conjunto de *Data Marts*. *Um Data Mart* (DM) pode ser definido como um *Data Warehouse* de pequena capacidade que abrange uma determinada área ou departamento, oferecendo informações mais detalhadas sobre um determinado assunto em questão.

O paradigma que Ralph Kimball defende baseia-se no facto da informação ser guardada utilizando o modelo dimensional (Kimbal, 1998). Outra das definições mais difundidas de DW foi dada por Bill Inmon que o descreve como "uma colecção de dados integrados, orientados por assunto, não voláteis, variáveis no tempo, e que fornecem suporte ao processo de apoio à decisão" (INMON, 1996). O paradigma deste autor difere um pouco do paradigma de Kimball. Bill Inmon defende que um *Data Warehouse* é apenas uma parte de todo o processo de *Business Intelligence*. Uma empresa possui um DW de onde os *Data Marts* extraem a informação, sendo a informação guardada na terceira forma normal. Ambos os paradigmas são válidos na medida em que representam filosofias diferentes de *Data Warehouse*.

No entanto, existem muitos e variados conceitos de DW dados por diferentes autores. Shams define *Data Warehouse* como uma plataforma que contém todos os dados da organização, centralizados e organizados para que os utilizadores possam extrair relatórios analíticos complexos, contendo informações para apoio à decisão (Shams, 2001).

Segundo o autor Devlin pode definir-se um *Data Warehouse* como um repositório de dados simples, completo e consistente, obtido de uma variedade de fontes e disponibilizado para o utilizador de maneira a que o entenda e o utilize no contexto organizacional (MALLACH, 2000).

Gardner define *Data Warehouse* por um processo, e não por um produto, para a montagem e administração de dados provenientes de várias fontes com o

propósito de obter uma visão simples e detalhada de parte de todo o negócio (Gardner, 1998).

Outra definição foi dada por Bigus que o define como um repositório de dados para uma grande quantidade de dados corporativos (MALLACH, 2000). Já Wang tem uma definição um pouco mais elaborada quando diz que o DW é o processo pelo qual os dados relacionados de vários sistemas operacionais são fundidos para proporcionar uma única e integrada visão de informação de negócios que abrange todas as áreas de uma empresa (COME, 2001).

Um DW permite reunir informações dispersas por diversas bases de dados e plataformas distintas, permitindo análises bastante eficazes, transformando dados dispersos em informações estratégicas que antes eram inacessíveis ou subaproveitadas (TAURION, 1997). Pode ser definido como um conjunto de componentes de *hardware* e *software* que podem ser usados para analisar uma quantidade massiva de dados que as organizações vão acumulando ao longo do tempo, de modo a apoiar na decisão e gestão do negócio. É portanto uma base de dados de suporte à decisão que é mantida separadamente dos sistemas operacionais da organização. Suporta processamento de informação fornecendo uma plataforma sólida para análise de dados históricos e consolidados.

As soluções de *Data Warehouse* incluem:

- Um repositório unificado e consolidado da informação da empresa, com um determinado histórico;
- Processos de extracção, transformação e carregamento dessa informação, proveniente de diferentes sistemas operacionais;
- Processos de "limpeza" de dados;
- Repositórios departamentais (*Data Marts*);
- Aplicações no cliente, para consulta à informação do repositório.

2.3 CARACTERÍSTICAS DE UM DATA WAREHOUSE

Um *Data Warehouse* deve (Kimbal, 1998) :

- Tornar as informações de uma organização acessíveis. O conteúdo do DW deve ser o mais compreensivo possível e o acesso aos dados deve ser efectuado com o melhor desempenho possível.
- Tornar as informações de uma organização consistentes. As informações provenientes de diferentes áreas da organização devem conseguir garantir a integridade semântica, ou seja, não devem existir problemas de nomes iguais para coisas diferentes, ou nomes diferentes para a mesma coisa.
- Ser uma fonte de informações flexível e adaptável. Deve ter a capacidade de suportar sucessivas modificações estruturais para inserção de novos dados, sem comprometer a estrutura já existente.
- Ser o alicerce para a tomada de decisão. Deve conter os dados na forma correcta para suportar a processo de apoio à decisão.

Para garantir estes requisitos, um DW deve seguir as características mencionadas em seguida (INMON, 1996).

2.3.1 ORIENTADO POR ASSUNTO

Uma das principais características do *Data Warehouse* é o facto de ter uma forte orientação por assunto. É organizado em torno de assuntos importantes, tais como por exemplo, cliente, produto e vendas. São por isso focados na modelação e análise de dados para quem toma decisões, em vez de operações diárias e processamento de transacções. Os *Data Warehouses* são seleccionados, isto é, fornecem uma visão simples e concisa sobre questões de um tema particular através da exclusão de dados que não são importantes no suporte ao processo de decisão. Pelo contrário, em ambientes operacionais as aplicações contêm dados necessários à satisfação imediata dos requisitos funcionais que podem ou não ser utilizados no processo de decisão. Os dados

do *Data Warehouse*, além de abrangerem um maior número de relacionamentos, têm em atenção um espectro temporal mais alargado.

2.3.2 Não Volátil

Um sistema operacional permite diversas operações de actualização dos dados (acrescentar, substituir, apagar). Num *Data Warehouse*, pelo contrário, só existem dois tipos de operações: uma introdução inicial dos dados e o acesso a estes, não requerendo por isso mecanismos de processamento de transacções, recuperação e controlo de concorrência. Os dados que são introduzidos no DW são estáticos, são dados que reflectem situações consolidadas, que não sofrerão actualizações. Os dados após serem extraídos, transformados e transportados para o DW estão disponíveis para os utilizadores apenas para consulta.

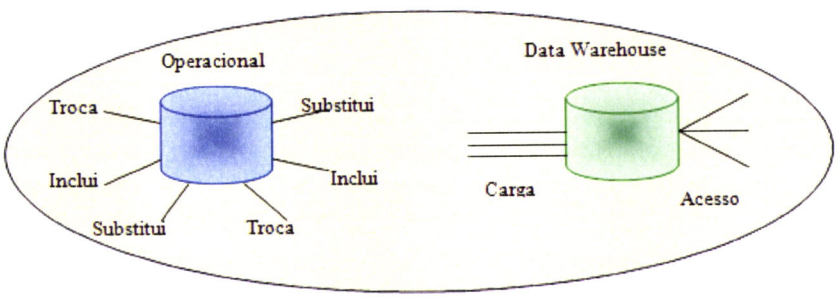

Figura 1- Não Volatilidade

2.3.3 Integrado

Um DW é construído por integração de múltiplas e heterogéneas fontes de dados. São aplicadas técnicas de limpeza de dados e integração de dados. A integração de dados, provenientes de sistemas operacionais, efectua-se nos mais variados níveis, na estrutura consistente de códigos, na forma consistente das variáveis, na conversão de nomes, etc. Os dados que são inseridos no DW devem estar consistentes entre si em termos de nomes, formatos e unidades de medida. Quando a informação é movida para o DW, é feita a conversão. No processo de integração dos dados, também pode ser necessário corrigir dados que estejam inconsistentes na origem, devido à não integração dos sistemas

transaccionais que fornecem os dados. Os dados fonte são modificados e convertidos para um estado uniforme de modo a permitir a carga no DW.

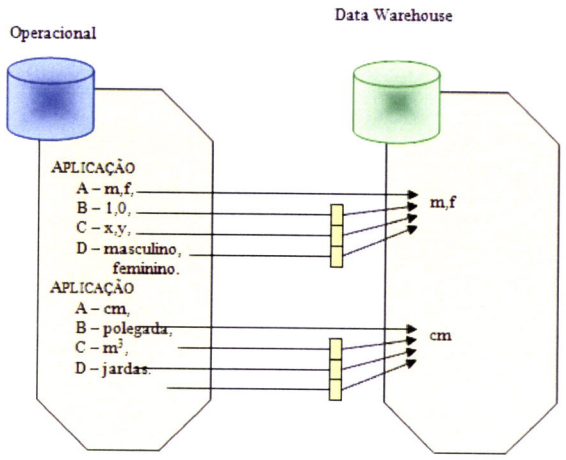

Figura 2 - Integração dos dados orientados por aplicação para um *Data Warehouse*.

2.3.4 HISTÓRICO

Num ambiente operacional os dados são exactos no momento em que são acedidos, ou seja, a base de dados operacional contem a informação actual. Um *Data Warehouse* tem geralmente diversas fontes de dados operacionais diferentes, o que significa que os dados aí acedidos podem já não ser os mais actualizados, pois podem já ter sido modificados no sistema operacional e o refrescamento do *Data Warehouse* ainda não ter sido efectuado. Assim, o *Data Warehouse* representa os dados sobre um horizonte de tempo distante (geralmente alguns anos) enquanto que o horizonte de tempo representado pelo âmbito operacional é mais curto (alguns dias).

Os dados no DW reflectem situações ao longo do tempo, ou seja, são dados históricos por natureza. Esta característica é essencial do ponto de vista de avaliação do negócio, pois é preciso manter o histórico do comportamento dos negócios para que seja possível analisar os resultados mais recentes. Os dados são regularmente inseridos para manter o registo histórico das operações de uma organização ao longo da sua existência. Existe portanto a noção de *timestamp* e de data de validade do registo. Isto permite que as consultas possam ser executadas para recuperar o estado da informação num dado

período de tempo. Os dados e os assuntos são sempre mutáveis, logo o DW terá que ter flexibilidade para acompanhar as mudanças de negócio. Não deve portanto ser alterado o histórico dos dados para não colocar em dúvida relatórios e análises já efectuadas.

A título de exemplo, na dimensão produto se um mesmo código de produto mudar de nome/descrição, deve criar-se um novo produto com o novo descritivo, e alterar no registo antigo a data final de validade:

- Antes da alteração:

CódigoProduto	Nome	DataInicioValidade	DataFimValidade	Válido
10	P1	01-01-2000		S

Tabela 1 – Exemplo de registo antes da alteração

- Depois da alteração:

Códigoproduto	Nome	DataInicioValidade	DataFimValidade	Válido
10	P1	01-01-2000	07-05-2008	N
10	P2	08-05-2008		S

Tabela 2 – Exemplo de registo depois da alteração

Todas estas características conduzem a um ambiente muito diferente do ambiente operacional. O *Data Warehouse* surgiu para dar resposta a novas necessidades de negócio e de concorrência empresarial, e não para substituir os sistemas operacionais em vigor. Existe uma certa redundância de dados entre estes dois ambientes, sendo de destacar:

- Os dados são filtrados quando passam do ambiente operacional para o *Data Warehouse*.
- Existe uma diferença temporal entre os dados dos dois ambientes.
- O *Data Warehouse* pode conter dados sumarizados. Para isso são construídas tabelas Agregadas derivadas de tabelas do *Data Warehouse*. Estas tabelas contêm informação "resumida" e agregada da forma pretendida de modo a optimizar e generalizar o processo de consulta

aos dados. Uma tabela com dados agregados ocupa menos espaço na base de dados do que uma tabela com dados mais pormenorizados.

- A transferência de dados sofre algumas transformações necessárias à integração dos dados.

2.4 ARQUITECTURA DO DATA WAREHOUSE

Entende-se por arquitectura, o conjunto de regras/estruturas a partir das quais é construído um sistema. Esta identifica e compreende o fluxo de dados através do sistema e a forma como serão utilizados dentro da própria organização. A estrutura de um *Data Warehouse* é constituída por uma base de dados independente, desenhada especificamente para apoio à decisão não podendo ser actualizada e à qual os utilizadores acedem através de uma ferramenta *front-end*. De entre os vários tipos de arquitectura, são de destacar os seguintes:

2.4.1 ARQUITECTURA TRADICIONAL DE DATA WAREHOUSING

Na arquitectura tradicional, os dados são integrados e carregados directamente para o *Data Warehouse,* sendo depois trabalhados de acordo com as funcionalidades do sistema e dos pedidos específicos dos utilizadores.

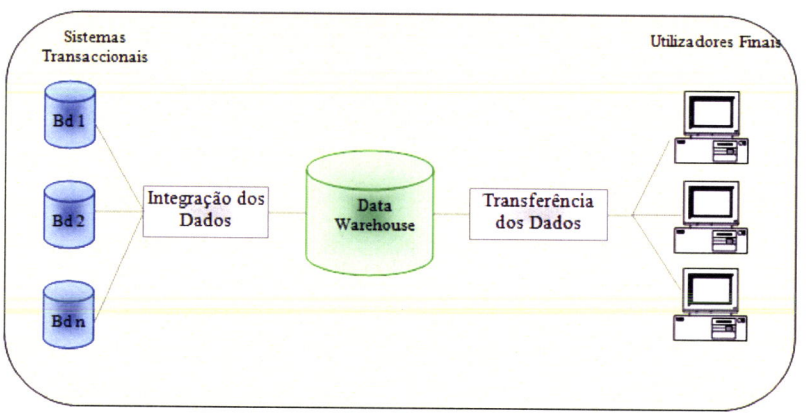

Figura 3 - Arquitectura Tradicional de *Data Warehousing*

2.4.2 Arquitectura em Três Camadas

Nesta arquitectura o *Data Warehouse* serve como depósito central dos dados históricos da organização. Esta camada é a base de alimentação dos *Data Marts*, orientados para uma dada área de negócio da empresa e compostos por conjuntos de tabelas estruturadas de forma a acelerar as respostas às consultas. Este modelo apresenta vantagens, tais como, o bom desempenho nas consultas, bons tempos na agregação dos dados e visão multidimensional entre estes. Constitui no entanto um problema para empresas que estão presentes em vários mercados e em diferentes países, pois terão que representar um universo mais complexo.

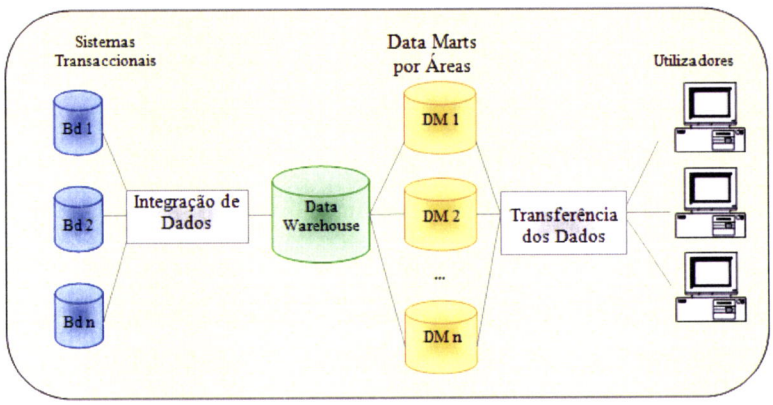

Figura 4 - Arquitectura em Três Camadas

2.4.3 Arquitectura em Duas Camadas

Este tipo de arquitectura não prevê a existência da camada correspondente ao *Data Warehouse*. Os *Data Marts* são actualizados directamente pelos sistemas transaccionais. Este modelo caracteriza-se pela alta velocidade de implementação.

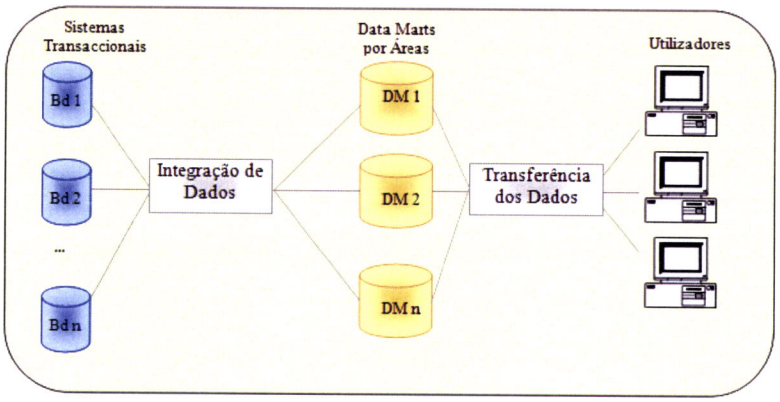

Figura 5 – Arquitectura em Duas Camadas

2.4.4 Base de Dados Integrada que Alimenta o Data Warehouse

Nesta arquitectura, os dados são extraídos, integrados e carregados dos diversos sistemas transaccionais da organização para uma base de dados integrada. Nesta base de dados, poderão ser ainda actualizados alguns dos dados através de uma aplicação. Este tipo de arquitectura é utilizado em organizações que começam por dar prioridade à tomada de decisões tácticas (curto prazo) no seu negócio. A integração prévia dos dados antes do seu carregamento no *Data Warehouse* é necessária em ambientes de grande turbulência onde o factor tempo é decisivo. A figura apresentada abaixo ilustra este tipo de arquitectura:

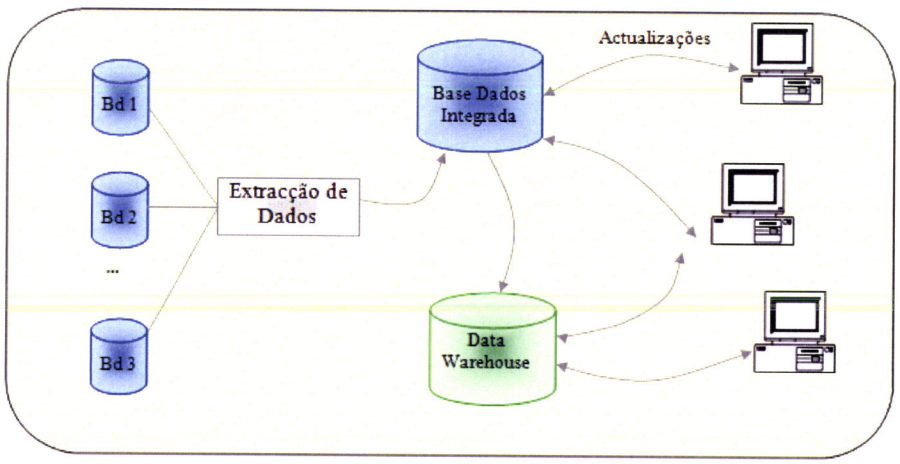

Figura 6 - Arquitectura em que o *Data Warehouse* é alimentado por uma BD integrada

2.5 Modelos de Arquitectura de Dados

Existem três modelos fundamentais de arquitectura de base de dados utilizados em sistemas de *Data Warehouse*:

- Estrela (*Star Schema*)
- Floco de Neve (*Snowflake*)
- Modelo Normalizado

2.5.1 Modelo em Estrela

Neste modelo, cada tabela Factual contem todos os níveis de agregação de dados possíveis. As chaves primárias existem apenas para os atributos mais atómicos (mais elementares). Cada tabela de *lookup* possui todos os atributos associados a essa dimensão. Este modelo é chamado modelo em Estrela porque a tabela de factos fica ao centro, cercada das tabelas de Dimensão, fazendo lembrar a forma de uma estrela. A vantagem deste modelo é que o número de operações de "join" entre tabelas é minimizado na generalidade das análises efectuadas. No entanto, a desvantagem deste modelo tem que ver com o facto das tabelas de Dimensão possuírem um tamanho muito grande.

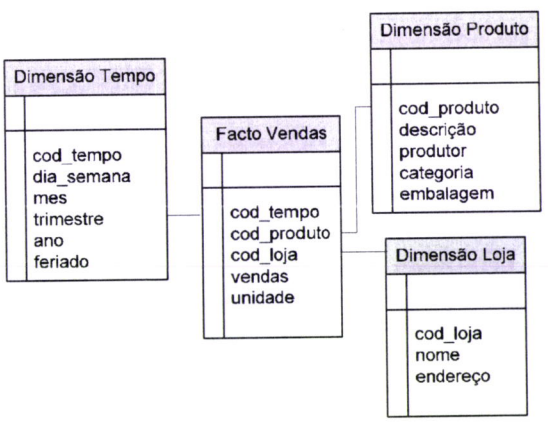

Figura 7 – Modelo em Estrela

2.5.2 Modelo Floco de Neve

No modelo de dados Floco de Neve, cada atributo de uma determinada dimensão é identificado por uma tabela de *lookup* diferente. Cada atributo incluí um ou mais atributos pais e cada nível de agregação de dados é

representado por uma tabela distinta. Neste caso, em vez do esquema em Estrela, passa a existir um conjunto de tabelas de Dimensão que se relacionam entre si através de alguns dos seus atributos. A vantagem deste modelo tem que ver com o facto das tabelas de atributos serem normalizadas, pequenas e sem redundâncias, diminuindo o espaço em disco. No entanto, as análises de níveis mais elevados implicam a realização de um grande número de operações de "*join*" entre tabelas, o que constituí uma desvantagem. Outra desvantagem é a sua manutenção complexa. Este modelo é chamado de Floco de Neve dado que cada dimensão "se divide" em várias outras tabelas organizadas de uma certa forma fazendo lembrar um floco de neve. A figura seguinte ilustra o aspecto de um modelo de dados Floco de Neve:

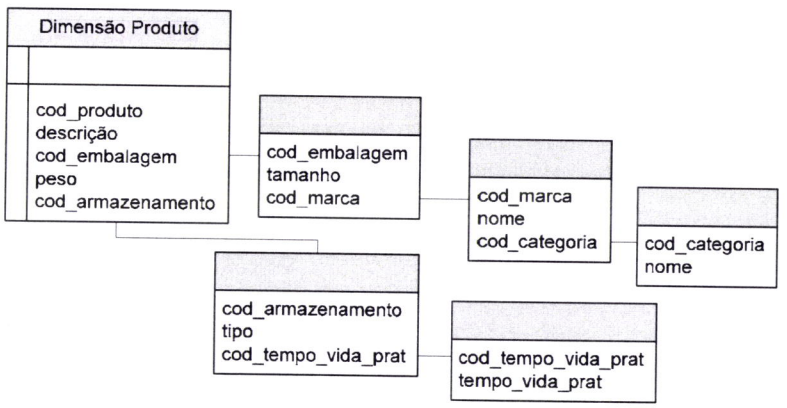

Figura 8 – Modelo Floco de Neve

2.5.3 Modelo Normalizado

Neste modelo, apenas os dados elementares ficam guardados em tabelas. Todas as agregações e acumulações de dados não ficam registadas. Se existirem, serão feitas em processos "*on-line*", em modo "*on-demand*" (ou seja, a pedido do utilizador).

2.6 Processos Básicos do Data Warehouse

Um *Data Warehouse* compreende vários sub-processos, tais como, extracção, transformação, carregamento e indexação, garantia de qualidade, publicação, consulta, segurança, recuperação e backup (Kimbal, 1998).

2.6.1 EXTRACÇÃO

O processo de extracção é o primeiro passo para obter a informação para o ambiente do *Data Warehouse*. Extrair significa ler e compreender fontes de informação, e copiar as partes necessárias, de relevante importância, para a área de estágio (estrutura intermédia entre os sistemas operacionais e o DW) para que futuramente essa informação seja trabalhada. A extracção de dados é feita muitas das vezes de sistemas legados. Os dados podem estar em diversos formatos (como por exemplo tabelas relacionais, arquivos ASCII, fontes Informix, etc), o que requer a sua tradução e adequação para a estrutura padrão do repositório.

2.6.2 TRANSFORMAÇÃO

A transformação de dados é efectuada com base em regras de negócio definidas com o objectivo de obter a informação mais importante para o negócio. A transformação dos dados em bruto envolve a adaptação, limpeza e consolidação da informação antes de ser integrada no DW. O objectivo principal desta etapa é eliminar as diferenças semânticas entre os dados extraídos e o esquema multidimensional adoptado. Extraída a informação para a área de estágio, existem variados possíveis passos de transformação, dentro os quais se podem destacar:

- Limpeza da informação, de modo a resolver conflitos existentes. Por exemplo, o nome de uma cidade que é incompatível com o seu código postal ou sobreposições, como por exemplo, dois campos representando a mesma informação com formatos diferentes.
- Exclusão de campos dos sistemas fonte que não são necessários no *Data Warehouse.*
- Complementação de campos com valor nulo, etc.

2.6.3 CARREGAMENTO E INDEXAÇÃO

Terminado o processo de transformação, a informação é carregada para o *Data Warehouse*. Devido ao grande volume de dados são utilizadas técnicas

especiais, como processamento paralelo ou incremental, para aumentar a eficiência e garantir o carregamento dos dados no DW na data prevista. Durante esta fase, outras actividades são desempenhadas, tais como a criação de índices, classificação e agregação de dados, particionamentos e verificação de integridade (para controlo de qualidade).

2.6.4 GARANTIA DE QUALIDADE

Quando cada *Data Mart* for carregado, o último passo antes que estes possam ser utilizados é a garantia de qualidade dos dados. Essa qualidade pode ser assegurada através da aplicação de diversas metodologias de qualidade e testes. Por exemplo, a execução de contagens sobre os dados carregados nas tabelas sendo que essas contagens terão que ser consistentes consoante os valores existentes nas fontes, a identificação de amostras de registos nas fontes e nas tabelas carregadas, a validação de mapeamentos, etc.

2.6.5 PUBLICAÇÃO

Quando cada *Data Mart* tiver sido prontamente carregado e tiver sido assegurada a qualidade dos dados, a nova informação está pronta para ser utilizada, sendo os utilizadores alertados para esse facto.

2.6.6 CONSULTA

A consulta aos dados não tem lugar na área de estágio. É realizada no *Data Warehouse*, e é o ponto fulcral para o seu uso. Inúmeros autores (MOODY, 2000), (WATTERSON, 1998), (Kimbal, 1998), (BREITNER, 1997), (POWER, 2000), argumentam que a solução mais viável é a divisão do DW em subconjuntos de dados significativos, que são alimentados consoante a área a que dizem respeito. Estes mini *Data Warehouses* são denominados *Data Marts*. O termo consulta envolve todas as actividades relacionadas com o acesso à informação num *Data Warehouse*.

2.6.7 Recuperação e Backup

Um *Data Warehouse* pode ser visto como uma corrente de dados dos sistemas legados para os *Data Marts* e eventualmente para os *desktops* dos utilizadores. De onde retirar a informação necessária com o propósito de ser arquivada ou recuperada de "desastres" é uma questão que merece bastante atenção. Mais complicado ainda é recuperar metadados que controlem o *Data Warehouse*.

2.6.8 Segurança de acesso ao Data Warehouse

A segurança é um tema que merece especial atenção, dada a importância dos dados armazenados no *Data Warehouse*. Apesar de existir a necessidade de disponibilizar a informação para o máximo de utilizadores possível, com interfaces de fácil utilização, existe também a necessidade de proteger a informação de *hackers*, *snoopers* e "espiões industriais". O desenvolvimento da *internet* tem vindo a amplificar drasticamente este dilema, dada a facilidade oferecida no acesso aos dados.

A segurança num *Data Warehouse* tem que ver, em grande parte, com o facto de reconhecer os utilizadores e fornecer-lhes direitos específicos de acesso aos dados. Nem todos os utilizadores têm acesso aos mesmos dados. É o *Data Warehouse Manager* quem qualifica os utilizadores e lhes fornece os respectivos direitos de acesso. O roubo de informação de um *Data Warehouse* poderia ser um alvo bastante lucrativo.

Alguns cenários típicos de segurança em *Data Warehousing* são por exemplo os seguintes:

- Uma empresa que controle um *Data Warehouse* que é usado por todas as suas áreas de negócio necessita de um sistema de segurança que assegure que os empregados de cada área apenas possuem permissões de acesso para aceder à informação que é relevante para a área em que actuam.
- Uma empresa pode armazenar no *Data Warehouse* informações pessoais. As leis de privacidade devem ser usadas para controlar o

acesso a essas informações pessoais. O acesso a estas leis de privacidade deve estar implementado no *Data Warehouse*.

- Uma empresa que venda informação contida num *Data Warehouse* aos seus clientes deve ter a capacidade de apenas permitir o acesso aos dados que cada cliente comprou. Nunca deve ser possível que os clientes vejam a informação de outros clientes, principalmente clientes que são concorrentes entre si.

Os privilégios de acesso, asseguram que um utilizador apenas pode efectuar uma operação sobre um objecto da base de dados se estiver autorizado a fazer essa operação. Um privilégio é uma autorização para permitir uma determinada operação. Por exemplo, sem os privilégios de *GRANT* um *user* não poderá aceder a determinada informação numa base de dados.

Os privilégios de sistema autorizam por exemplo um *user* a efectuar uma operação específica de *CREATE TABLE*, que permite a criação de uma tabela na base de dados. Os privilégios a objectos autorizam um *user* a efectuar uma determinada operação sob um determinado objecto.

De acordo com Ralph Kimball, existem algumas tecnologias de segurança importantes que devem ser utilizadas num *Data Warehouse*, sendo de destacar as seguintes: *Routers e Firewalls*, *Directory Server*, Encriptação, *Signed Software Certificates* e *Secure Sockets Layer*.

2.6.8.1 ROUTERS E FIREWALLS

Possuem a capacidade de rejeitar tentativas de conexão por parte de *hosts* desconhecidos (intrusos). Têm como objectivo aplicar uma politica de segurança a um determinado ponto de controlo da rede. São portanto um mecanismo de segurança bastante importante num DW.

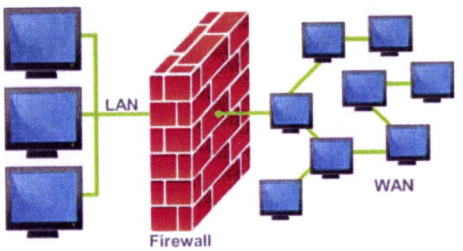

Figura 9 – Firewall

2.6.8.2 Directory Server

É como que um *Data Warehouse* de recursos disponível na rede. Esses recursos envolvem máquinas, repositórios de documentos, sistemas de transacção, impressoras, nomes e endereços, contas de e-mail, etc. Deve ser o ponto central e singular para autenticação e autorização de todos os *users* do *Data Warehouse* independentemente se estão conectados à rede interna ou se são provenientes do exterior (Internet). Esta tecnologia pode ser vista como uma consola de administração onde as politicas de segurança são definidas. O DBA (*Data Base Administrator)* pode correr máquina a máquina e definir os privilégios aos utilizadores. Para a implementação do *Directory Server* é utilizado o protocolo LDAP (*Lightweight Directory Access Protocol*), protocolo derivado do X. 500.

2.6.8.3 Encriptação

É a técnica de trocar o conteúdo de uma mensagem compreensível (dentro de um domínio de onde fazem parte os membros da comunicação) por um conteúdo incompreensível (texto encriptado), através de uma função de transformação. A forma mais simples de encriptação é a encriptação simétrica. Na encriptação simétrica a chave utilizada para encriptar os dados é a mesma chave que é usada para desencriptá-los. Este tipo de encriptação também é chamado de encriptação de chave privada. A figura abaixo ilustra este tipo de encriptação:

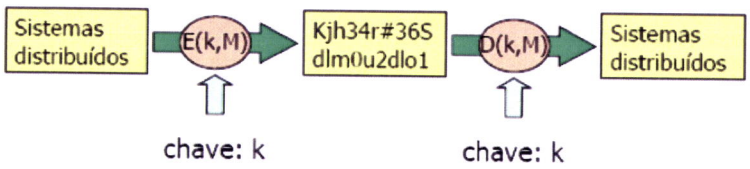

Figura 10 – Encriptação Simétrica

Um dos principais algoritmos de encriptação simétrica é o DES (*Data Encryption Standard*) que possui chaves com 56 bits. No entanto este tipo de encriptação apresenta alguns problemas. Se por exemplo um *user* encriptar informação com uma chave privada que não revela a ninguém, caso se esqueça dessa chave a informação será perdida para sempre. Dado o principal problema de como distribuir em segurança as chaves, surgiu a encriptação assimétrica, também conhecida como encriptação de chave pública. Nesta técnica, a chave usada para encriptar dados é diferente da chave usada para desencriptar. Este método baseia-se em chaves distintas e cada utilizador deve possuir um par de chaves, uma pública usada para encriptar dados e uma outra privada para desencriptar:

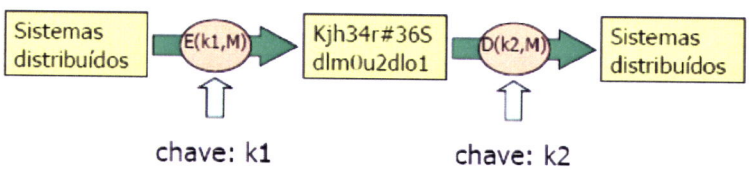

Figura 11 – Encriptação Assimétrica

2.6.8.4 Signed Software Certificates

Basicamente, neste tipo de mecanismo qualquer componente de *software* vem acompanhado de uma mensagem que identifica o criador do *software*. Essa mensagem está encriptada com a chave privada do criador. Só são recepcionadas componentes de *software* caso se possua a chave pública do criador do *software*. No entanto, no caso de não se possuir a chave, esta pode ser requerida ao certificado de segurança (*Trusted Certificate Authority*).

Depois de se verificar se o *software* é realmente de confiança pode escolher-se se este vai correr ou não no sistema.

2.6.8.5 Secure Sockets Layer

No ano de 1995, a Netscape (empresa famosa por ter produzido o conhecido *browser* de Internet *Netscape Navigator*) apresentou o *Secure Sockets Layer* (SSL) como um protocolo standart para comunicação entre cliente e servidor. Este protocolo é hoje em dia usado em muitos dos browsers de Internet. O SSL utiliza uma combinação entre encriptação de chave pública e encriptação de chave simétrica. A encriptação de chave pública é usada na máquina cliente para verificar se realmente foi estabelecida a ligação com a máquina servidor que era destinada. Opcionalmente, o servidor pode também exigir uma verificação da identidade do cliente. Esta ligação é chamada de *Handshake* e é utilizada para que as duas máquinas cheguem a acordo relativamente à partilha da chave de encriptação simétrica. Esta chave de encriptação simétrica é utilizada para as comunicações de alta velocidade que são exigidas. Pode ser alterada pelas duas máquinas a cada segundo que passa, com o objectivo de manter elevado o grau de segurança.

3 Conceitos Envolvidos

3.1 Conceito de Business Intelligence

O conceito de *Business Intelligence* (BI) não é recente. Fenícios, persas, egípcios e outros povos do Oriente já utilizavam este princípio há milhares de anos, quando cruzavam informações provenientes da natureza. Observar e analisar o comportamento das marés, os períodos de seca e de chuvas, a posição dos astros, entre outros, eram formas de obter informações que eram utilizadas para tomar decisões que permitissem uma melhoria de vida nas comunidades. O mundo mudou desde então, mas o conceito permaneceu o mesmo. O interesse pelo BI tem vindo a crescer na medida em que a sua utilização possibilita às empresas realizar análises e simulações, de forma a tornar mais eficientes os processos relacionados com o apoio à decisão.

Este termo surgiu nos anos 80 e descreve a capacidade que as empresas têm de aceder a dados e explorar as informações, analisando-as e desenvolvendo percepções a seu respeito. Com o passar dos anos o termo *Business Intelligence* ganhou maior abrangência, dentro de um processo natural de evolução, envolvendo uma série de ferramentas, como o próprio EIS (*Executive Information System*), as soluções de DSS (*Decision Support System*), geradores de consultas e de relatórios, *Data Marts*, *Data Mining*, ferramentas OLAP (*On-Line Analytical Processing*), entre outras. É um conceito "*up dated*" que vai além da gestão empresarial.

Em soluções de *Business Intelligence* para empresas de telecomunicações, é necessário que a informação chegue de forma rápida e coerente porque a sobrevivência no mercado será medida pela capacidade de "gerar conhecimento". O retorno que se espera de um sistema de BI depende das prioridades de cada empresa. As ferramentas de BI continuam e continuarão a evoluir dado que este mercado possui um enorme potencial de crescimento.

3.1.1 Arquitectura de um Sistema de BI

Um sistema típico de BI é composto pelas seguintes componentes:

- ETL: Componente que se dedica à extracção, carga e transformação de dados. É a parte responsável pela recolha de informações provenientes de diversas fontes.
- *Data Warehouse*: Local onde ficam concentrados os dados extraídos dos sistemas operacionais. A vantagem de ter um repositório de dados à parte é a possibilidade de armazenar informações históricas e agregadas, dando um melhor suporte para as análises futuras.
- *Front-end*: É a parte visível ao utilizador de um projecto de BI. Pode ser em forma de relatórios padronizados e *ad hoc*, portal de intranet / internet / extranet, análises OLAP entre outras funções, como *Data Mining* ou simulações futuras.

Figura 12 – Arquitectura de um sistema de BI

3.2 EIS vs DSS

O *Executive Information System* foi inventado pelo Instituto de Tecnologia de Massachusetts (MIT) nos finais dos anos 70 e início dos anos 80. Um EIS acede a dados sobre os factores críticos de sucesso de uma empresa e permite aos executivos o acesso à informação de um modo muito interactivo e flexível. Os EIS são muitas vezes confundidos por algumas pessoas com os Sistemas de Suporte à Decisão (DSS).

O domínio de um EIS estende-se geralmente ao dos relatórios e ao da informação presente e passada. Os DSS podem cobrir não só estas áreas, mas

também fazer projecções futuras e acrescentar funções de análise e de diagnóstico. A figura abaixo pretende ilustrar os domínios destes sistemas:

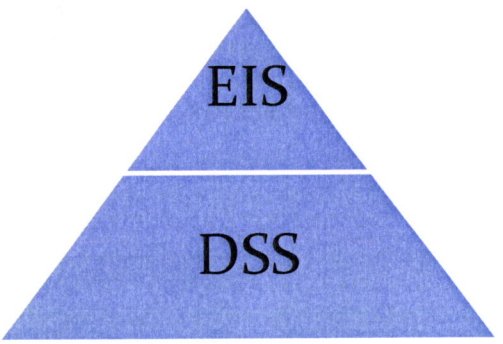

Figura 13 – Pirâmide EIS vs DSS

O objectivo de um EIS é fornecer o acesso rápido à informação através de uma interface amigável, aos executivos de alto nível. A grande vantagem é que quem utiliza estes sistemas não terá que possuir conhecimentos detalhados de informática. Muitas vezes um executivo não tem muita afinidade com computadores e também não quer ter, porque não precisa. O EIS também pode permitir que o utilizador altere o nível de detalhe da informação com a ajuda de uma ferramenta OLAP. Com o uso integrado desta ferramenta o utilizador poderá transformar as informações que se encontram na forma de tabelas em gráficos ou vice-versa, como também poderá executar um *Drill-Down* ou um *Drill-Up* para detalhar ou resumir o grau de granularidade da informação.

Um EIS terá que possuir uma base de dados para extracção de informação. Essa base de dados pode ser construída utilizando os vários sistemas transaccionais da empresa. Porém o tempo de espera da conclusão dessas transacções pode ser algo moroso, dada a necessidade de extrair dados de diversas fontes, trata-los, mantê-los íntegros bem como um histórico dos mesmos. Então, para a aplicação de um EIS deve ser utilizado um *Data Warehouse* permitindo que o EIS tenha acesso a essas informações após estarem devidamente tratadas.

No sentido funcional, os EIS e DSS são diferentes mas complementares, e não limitativos quanto aos seus domínios e objectivos:

EIS:

- Fornecem apoio aos gestores na identificação de problemas e oportunidades.
- Centrados no fácil e rápido acesso a informações internas e externas.
- Grande utilização de interfaces gráficas amigáveis.
- Geralmente focados no nível estratégico.

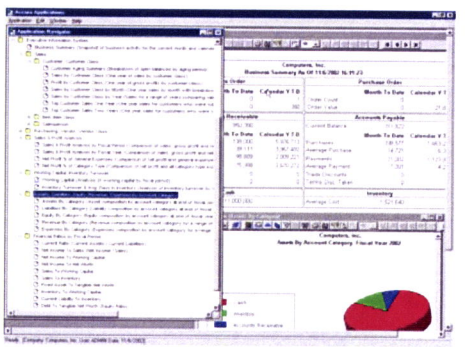

Figura 14 – Aspecto gráfico de um EIS (painel de controlo)

DSS:

- Análise de problemas ou oportunidades mais específicos a uma dada área.
- Análises ad hoc.
- Geralmente focados no nível táctico e estratégico.
- Simulações, cálculos, consultas.
- Análises *What-if*

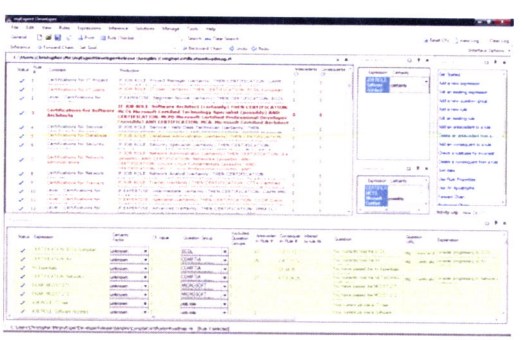

Figura 15 – Aspecto gráfico de um relatório gerado por DSS

A título de exemplo, um sistema de EIS pode reportar a necessidade de uma determinada acção de marketing para ganhar vantagem de mercado. Com o auxílio de um DSS (simulações e outras ferramentas), o gestor decide qual a decisão a tomar.

Muito se tem discutido sobre a conceituação destes sistemas. No inicio foi um pouco relutante a aceitação da sigla EIS como um novo conceito, pois muitos viam estes sistemas como sendo uma adaptação e/ou apenas mais uma ferramenta complementar de um DSS. Mas não demorou muito para que fossem aceites como um novo conceito, dadas as suas características e forma de implementação.

3.3 BICC

O conceito de BICC (*Business Intelligence Competency Center*) é um conceito recente. Um BICC consiste num conjunto de pessoas de diversas áreas, como por exemplo analistas de negócio, especialistas em tecnologias de informação, gestores, a trabalharem em conjunto pelo objectivo comum de garantir uma evolução apropriada em BI. Basicamente é um grupo formado por pelo menos um representante de cada departamento chave da empresa. Este grupo é focado numa única estratégia de negócio e reúne todos os dados importantes, analisando-os e utilizando-os para gerar resultados para toda a empresa. O centro desenvolve um plano estratégico total, as prioridades para o BI, define as exigências e ajuda a empresa a interpretar e aplicar o "insight" às decisões de negócio.

Na prática, estudos mostram que com um BICC a empresa consegue aumentar a precisão das decisões, melhorar a colaboração entre as áreas de negócio e de tecnologia, bem como aumentar a velocidade no apoio à decisão. Acredita-se que em 2009, mais de 60% das empresas e agências governamentais com estratégias de BI utilizarão um BICC. Definir e suportar um BICC pode ser a chave para assegurar que uma organização ganhará o máximo nos seus investimentos.

3.4 DATA WAREHOUSE VS DATA MART

Estes dois conceitos são muito importantes e há que fazer uma distinção entre eles. *Um Data Mart* pode ser definido como um *Data Warehouse* de menor capacidade, que abrange uma área ou departamento específico, oferecendo informações mais detalhadas sobre um determinado assunto. É por isso uma abordagem específica do DW e o seu domínio abrange apenas uma área específica da empresa. Um *Data Warehouse* pode ser visto como um conjunto de *Data Marts*, contendo todas as informações da empresa provenientes de diversas fontes de dados operacionais, dispostas de forma integrada e consolidada.

Existem algumas opiniões divergentes sobre este tema. Segundo Inmon os *Data Warehouses* e os *Data Marts* têm estruturas essencialmente diferentes. Na perspectiva deste autor, é difícil integrar um conjunto de *Data Marts* e mesmo que se conseguisse não resultaria num *Data Warehouse*. O *Data Mart* resulta do *Data Warehouse* (INMON, 1996). Já Kimball tem uma opinião completamente diferente, indicando que o *Data Warehouse* é constituído pela união de todos os seus *Data Marts* (Kimbal, 1998).

3.5 STAGING AREA

Os processos mais importantes na formação de um *Data Warehouse* são sem dúvida os que envolvem a *Staging Area* (área de estágio). Esta constitui uma área intermédia de armazenamento da informação entre os sistemas operacionais e o *Data Warehouse*. Os sistemas operacionais são fontes de dados (normalmente ainda em "bruto") que abastecem o DW. Os dados provenientes desses sistemas operacionais necessitam de ser preparados, para que sejam carregados no *Data Warehouse*. Para isso, são utilizadas as tabelas de *Staging Area*, que são responsáveis por este armazenamento intermédio de dados. É nesta fase que é implementado o processo de ETL (Extracção, Transformação e Carregamento), essencial no *Data Warehouse*.

3.6 ETL

3.6.1 EXTRACÇÃO

O objectivo dos métodos de extracção é isolar os dados que serão utilizados pelos sistemas de apoio à decisão. É necessário filtrar apenas os dados que serão necessários, a fim de se evitar desperdício de desempenho e de espaço.

Existem diversas técnicas de extracção de dados. Os dados podem ser extraídos periodicamente, sendo que neste caso é o utilizador que define qual a periodicidade do processo. Assim que é inicializado, o processo extrai as modificações realizadas desde o período da última actualização, actualizando os dados no DW.

Uma vantagem desta técnica é a possibilidade de execução em horários fora de trabalho, não comprometendo por isso o desempenho dos sistemas transaccionais. Muitas vezes são utilizadas as chamadas tabelas "snapshot" que são cópias de tabelas dos sistemas operacionais carregadas para o DW. No entanto, os dados podem também ser extraídos de modo incremental, ou seja, este método tem como objectivo extrair todos os estados pelos quais os registos passam, pelo que qualquer alteração de valores é identificada pelo sistema e actualizado o DW. Esta técnica acaba por armazenar todo o histórico de registos.

3.6.2 TRANSFORMAÇÃO

A partir do momento em que os dados já tenham sido extraídos dos sistemas operacionais para a área de estágio, a fase de transformação é iniciada. Nesta fase, um conjunto de processos é iniciado, tais como limpeza, atribuição de novas chaves, etc. As chaves que identificam as dimensões do *Data Warehouse* não devem ser as mesmas existentes nos sistemas operacionai, de modo a acompanhar as actualizações nas dimensões (Kimbal, 1998). As rotinas de limpeza e integração dos dados têm como objectivo assegurar a consistência no DW. Por vezes são realizadas exclusões de informações desnecessárias, excluindo atributos e entidades que não estejam no âmbito a ser tratado pelo

DW. São também por vezes excluídas relações entre tabelas que influenciam o desempenho das consultas ou efectuado o "merging" de tabelas. É sempre adicionada a dimensão Tempo, dada a necessidade de identificar possíveis modificações nos dados com o decorrer do tempo. Isto vai possibilitar a realização de análises comparativas sobre os factos, com o objectivo de manter armazenado o histórico dos dados ao longo do tempo. É justamente esta dimensão que vai determinar o grau de granularidade do DW, como por exemplo, dados relativos a um dia, uma semana, um mês, ou até mesmo um ano.

3.6.3 CARREGAMENTO

O processo de carregamento é realizado após todos os tratamentos efectuados aos dados nos processos de extracção e transformação. Esta etapa consiste em carregar os dados tratados e armazenados na *Staging Area* e migrá-los para o *Data Warehouse*.

3.7 ODS

O conceito de ODS *(Operational Data Storage)* surgiu por volta dos anos 90, e era visto como sendo um tipo de *Data Warehouse*. O ODS é formado por dados retirados dos diversos sistemas operacionais e sujeitos a diversas operações de transformação e conversão. Contem informação de detalhe de cariz transaccional, actual e sujeita a processos de actualização regular. Permite obter uma visão global, integrada e actual dos principais dados de produção. Fornece suporte ao processo de tomada de decisões detalhadas, de cariz operacional, exigindo respostas quase imediatas. O seu histórico é curto (algumas semanas) e a informação está organizada por áreas de análise.

Segundo Ralph Kimball, o ODS é uma parte do ambiente do *Data Warehouse* que para muitos analistas ainda gera alguma confusão sobre o seu conceito. De acordo com Bill Inmon e Claudia Imhoff, no livro *"Building the Operational Data Store"*, o ODS é uma estrutura de armazenamento de dados recentes, orientada ao assunto, integrada, volátil e que contém apenas dados

corporativos detalhados (William H. Inmon, 1996). Inmon refere que o ODS é um sistema operacional separado do *Data Warehouse*, que suporta constantes acessos e actualizações operacionais. No entanto, Ralph Kimball tem uma opinião um pouco diferente, referindo que o ODS faz parte do *Data Warehouse*, contendo dados integrados com granularidade mais detalhada, podendo ser utilizado para dar suporte ao nível mais baixo do DW. Segundo este autor, o ODS funciona como um ponto de integração dos sistemas operacionais, sendo uma fonte de informação detalhada em tempo real. É como que o "*front edge*" do *Data Warehouse.*

Ralph Kimball concorda também que o ODS deve ser uma estrutura de dados orientada ao assunto, integrada, frequentemente actualizada com informação detalhada para suportar os sistemas transaccionais. Quaisquer dados detalhados usados no apoio à decisão devem simplesmente ser vistos como o nível mais baixo do *Data Warehouse* (Kimbal, 1998).

O ODS pode ser visto como uma "réplica" da base de dados operacional, de onde é possível efectuar análises e extrair relatórios sem afectar em nada a base de dados "principal".

3.8 OLAP

No fim da década de 60 a IBM desenvolveu uma linguagem chamada APL. Esta linguagem foi a primeira a utilizar análise multidimensional, um termo fundamental em OLAP, sendo muito utilizada nas décadas de 80 e 90 em aplicações de negócio. Na década de 90 surgiu uma nova classe de ferramentas, designada OLAP. Estas ferramentas utilizam a maioria dos conceitos da linguagem APL.

OLAP significa *On-Line Analytical Processing* e é um mecanismo de manipulação de dados de alta capacidade, destinado a suportar e operar sobre estruturas de dados multidimensionais.

O termo OLAP foi usado pela primeira vez por E. F. Codd, o qual também definiu doze regras para as aplicações OLAP. A ideia de visão multidimensional é uma das doze regras e tornou-se numa característica

fundamental da tecnologia OLAP. A capacidade de se puder observar uma base de dados no formato de um cubo, contendo duas, três, quatro ou até mais dimensões, permite dividi-lo em qualquer uma das suas dimensões. Uma dimensão é uma unidade de análise com dados agregados. Por exemplo, a dimensão Tempo poderia ter os dados agregados em dias, meses, anos, etc. A dimensão Local poderia ter os dados agrupados por bairro, cidade, estado, país, etc. A Modelação dimensional permite assim analisar um negócio em qualquer uma das suas dimensões (visões de negócio). A figura abaixo permite que se tenha uma ideia da visão multidimensional dos dados:

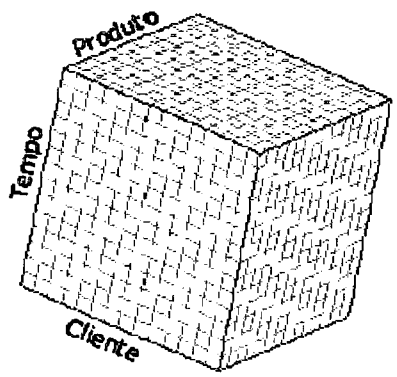

Figura 16 – Modelação dimensional

Esta tecnologia permite analisar e visualizar os dados de uma empresa de forma rápida, consistente e principalmente interactiva.

A tecnologia OLAP trouxe uma grande capacidade de efectuar cálculos complexos, tais como previsões percentuais de crescimento e médias diversas, sendo considerada a variável tempo. É uma ferramenta muito importante no contexto empresarial, ajudando a analisar de forma mais eficiente a quantidade de dados crescente que é armazenada pelas empresas, transformando-os em informação útil (THOMSON, 2002).

A tecnologia OLAP é geralmente implementada num ambiente multi-utilizador e cliente/servidor, oferecendo assim respostas rápidas às consultas *ad-hoc* (construção de listagens, interligando a informação disponível na base de dados conforme as necessidades especificas da empresa, assim como a sua exportação, possibilitando várias simulações). Hoje em dia, esta tecnologia

também vem sendo disponibilizada no ambiente *Web*. Auxilia o utilizador a sintetizar informações corporativas através de visões comparativas e personalizadas, análises históricas e projecções.

É importante distinguir os termos OLAP e OLTP (*On-Line Transaction Processing*). A tecnologia OLTP é utilizada no processamento de transacções em bases de dados relacionais. A tabela abaixo resume a diferença entre estas tecnologias:

	OLTP	OLAP
Operação Típica	Transacção	Análise
Granularidade	Atómico	Agregado
Temporalidade dos dados	Presente	Histórico, actual e futuro
Recuperação	Poucos registos	Muitos registos
Utilizadores	Muitos	Poucos
Orientação	Registos	Arrays
Consulta	Predefinida	Ad-hoc

Tabela 3 – Diferenças entre OLTP e OLAP

No modelo OLTP, como operação típica são efectuadas transacções, que podem ser actualizações de registos, uma remoção, uma recuperação, ou até mesmo, uma criação de registos. Já no modelo OLAP os dados servem para ser analisados, por exemplo, saber qual foi o produto mais vendido há dois meses atrás.

Os dados são tratados com o máximo de detalhe no modelo OLTP, sendo que no modelo OLAP os dados estão agregados, ou seja, há um resumo dos dados para que sejam feitas as análises.

Na tecnologia OLTP o que importa são os dados actuais, por exemplo, se o preço de um produto passa de x para y, então x é apagado e substituído por y. No caso da tecnologia OLAP o passado é registado, pois os dados são utilizados para análise e previsões futuras, sendo necessário que fiquem registadas informações sobre o passado.

No modelo OLTP geralmente poucos registos são recuperados numa consulta, por exemplo, o preço de um produto ou os dados de um cliente. Na tecnologia OLAP vários registos são recuperados, por exemplo, a quantidade vendida de um dado produto em cada um dos últimos dois meses agrupados por cidade.

No modelo OLTP existem muitos utilizadores a utilizar o sistema ao mesmo tempo, porém com consultas mais simples. No OLAP existem poucos utilizadores mas com consultas mais complexas.

A nível de orientação, o modelo OLTP é orientado a registos. O modelo OLAP utiliza arrays para representar dimensões, como por exemplo, a dimensão Tempo ou a dimensão Lugar.

No modelo OLTP os tipos de consultas que podem ser realizadas já são predefinidas, como por exemplo, consultar o preço de um produto ou consultar o débito de um cliente. No modelo OLAP as consultas são *ad hoc*, ou seja, são definidas de acordo com os interesses do utilizador que realiza a consulta, por exemplo, qual o total de vendas de um determinado produto em cada um dos últimos três meses, agrupado por cidade.

3.8.1 Características das ferramentas OLAP

- *Drill Across:* ocorre quando o utilizador "salta" um nível intermédio dentro da mesma dimensão. Por exemplo, a dimensão Tempo é composta por ano, semestre, trimestre, mês e dia. A operação *Drill Across* é executada quando o utilizador passa de ano para trimestre ou mês.
- *Drill Down:* ocorre quando o utilizador aumenta o nível de detalhe da informação, diminuindo a granularidade. A granularidade determina quais os tipos de consultas que podem ser feitas no DW. Influência directamente a velocidade de acesso à informação e o volume de dados armazenados.
- *Drill Up:* é o contrário do *Drill Down*, ocorre quando o utilizador aumenta a granularidade, diminuindo o nível de detalhe da informação.

- *Drill Throught:* ocorre quando o utilizador passa de uma informação contida numa dimensão, para uma outra. Por exemplo, inicia-se na dimensão tempo e no próximo passo analisa a informação por região.
- *Slice and Dice:* é uma das principais características de uma ferramenta OLAP. Como a informação é armazenada em cubos multidimensionais, surgiu a necessidade de criar este módulo. É responsável por trabalhar a informação que está armazenada multidimensionalmente. É utilizado para modificar a posição da informação, como por exemplo, trocar linhas por colunas para facilitar a compreensão dos utilizadores e girar o cubo sempre que houver necessidade.

3.8.2 ARQUITECTURA DAS FERRAMENTAS OLAP

Existem vários tipos de arquitecturas OLAP, sendo de destacar as seguintes:

- MOLAP *(Multidimensional On Line Analitical Processing)*
- ROLAP *(Relational On Line Analitical Processing)*
- HOLAP *(Hybrid On Line Analitical Processing)*

3.8.2.1 MOLAP

Na arquitectura MOLAP (OLAP multidimensional) os dados ficam armazenados numa base de dados multidimensional, sendo armazenados num espaço menor do que aquele que é necessário para armazenar os mesmos dados numa base de dados relacional. Os dados são mantidos em estruturas de dados do tipo array, para permitir uma maior velocidade de acesso, sendo armazenados com uma granularidade específica.

3.8.2.2 ROLAP

A arquitectura ROLAP (OLAP relacional) é uma simulação da tecnologia OLAP feita em bases de dados relacionais, possuindo a vantagem de não restringir o volume de armazenamento de dados. Esta ferramenta não utiliza cubos pré-calculados como a arquitectura MOLAP. É mais tolerante às

alterações nas fontes de dados originais quando são OLTP, dado que a arquitectura ROLAP também utiliza OLTP.

3.8.2.3 HOLAP

A arquitectura HOLAP (OLAP híbrido) é uma mistura de arquitectura ROLAP com MOLAP. A ideia é armazenar os dados de maior granularidade do DW em estruturas relacionais normalizadas e os agregados de menor granularidade em estruturas dimensionais, tais como cubos OLAP. Esta arquitectura consegue combinar a capacidade e a escalabilidade das ferramentas ROLAP com o bom desempenho das bases de dados multidimensionais (THOMSON, 2002).

3.9 METADATA

A definição mais comum de metadata é "dados sobre dados". Devido ao grande volume de dados contidos no *Data Warehouse* é necessário que exista uma forma flexível e eficiente de acesso aos dados. É necessário saber que dados estão disponíveis e onde estão localizados. A metadata pode ser vista como a descrição dos dados, do seu ambiente, como são manipulados e para onde são distribuídos.

A metadata permite definir as estruturas de informação usadas, os algoritmos utilizados para a transformação, conversão, acumulação e agregação de dados. Permite identificar as fontes de informação, bem como qual o destino dos dados. Permite também controlar o mapeamento de dados do ODS para o DW (relações existentes entre campos e atributos nos arquivos fonte e nos arquivos destinatários). A metadata pode servir também para fazer a consolidação dos termos e temas dos dados. Um determinado termo deve ter o mesmo significado em todo o *Data Warehouse*.

Segundo Inmon, a metadata mantém informações sobre "o que está onde". De acordo com este autor, as informações que a metadata mantém são as seguintes (INMON, 1996):

- A estrutura dos dados segundo a visão do programador;

- A estrutura dos dados segundo a visão dos analistas;
- A fonte de dados que alimenta o DW;
- A transformação efectuada aos dados no momento da sua migração para o DW;
- O modelo de dados;
- O relacionamento entre o modelo de dados e o DW;
- O histórico das extracções de dados;
- Os dados referentes aos relatórios que são gerados pelas ferramentas OLAP, assim como os que são gerados nas camadas semânticas.

Existem algumas perguntas que são respondidas pela metadata, como por exemplo:

- Que tabelas, atributos e chaves contém o *Data Warehouse*?
- Qual a origem de cada conjunto de dados?
- Com que frequência são carregados os dados?
- Qual o volume de dados existente?

3.10 NEAR REAL TIME DATA WAREHOUSE

Os *Data Warehouses* tradicionais não contêm informação em tempo real, ou seja, informação sobre o dia actual. A informação que é obtida dos sistemas operacionais diz respeito a dias anteriores ao actual. Dadas as exigências do mundo actual, de ter acesso à informação mais actualizada possível, surgiu o *Near Real Time Data Warehouse*. Por exemplo, as agências governamentais e aeroportos necessitam de ter a capacidade de puder analisar a informação mais actual possível, dado que pretendem detectar grupos suspeitos de passageiros ou até mesmo actividades ilegais que possam estar a decorrer.

Um *Real Time Data Warehouse* (RTDW) suporta uma entrega de dados contínua e assíncrona. A informação provém directamente da fonte sem a necessidade de área de estágio. Esta acção é efectuada após a informação original ser escrita. Qualquer atraso na recepção da informação deve-se unicamente à latência que pode existir no transporte dos dados e

(opcionalmente) aos tempos mínimos de processamento para tratar ou transformar a informação a ser entregue.

O que é interessante num RTDW é o facto de a informação ser capturada continuamente em tempo real. Ficam registadas todas as alterações significativas que ocorrem nos dados à medida que vão surgindo. Isto elimina uma das mais "incómodas" partes de todo o processo de *Business Intelligence*, que é detectar e extrair as alterações que ocorrem nos sistemas operacionais. O RTDW elimina algumas falhas que possam ocorrer no acesso à informação que é disponibilizada. O processamento contínuo sem *delay* (atrasos de tempo) abre novas e significativas oportunidades para a prática do *Business Intelligence*.

4 Importância de um Data Warehouse para uma organização de Telecomunicações num ambiente em constante mutação

Nos últimos anos, a indústria das telecomunicações tem sofrido um incrível crescimento e desenvolvimento. Este crescimento afectou todas as linhas de negócio das telecomunicações, e com o evoluir do tempo a estrutura das empresas tem vindo a sofrer grandes alterações, tornando-se necessário dar respostas mais rápidas e eficientes às necessidades dos consumidores. Isto para que a competitividade continue num ambiente em constante mutação. A capacidade de responder de forma flexível às rápidas mudanças de ambiente tornou-se um factor essencial para o sucesso das empresas. Que empresas vão "sobreviver" e como será o futuro das empresas de telecomunicações, são questões que só terão resposta dentro de alguns anos. (Matisson, 1997)

A indústria das telecomunicações tem influenciado significativamente o crescimento do *Data Warehousing* nas últimas duas décadas (PETERS, 2000). O *Data Warehousing* pode ajudar a estimar o sucesso que terá um novo produto durante o seu ciclo de vida. Pode também permitir saber como vai o cliente aceitar esse novo produto, bem como pode ajudar a criar simulações de preços e ganhos de receita. É muito importante saber o que manter armazenado e por quanto tempo.

O que é interessante sobre o fenómeno do *Data Warehousing* é que a indústria de telecomunicações tem vindo a tornar-se a força pioneira desta tecnologia. É possível por exemplo através de um *Data Warehouse*, estimar a probabilidade das pessoas mudarem de fornecedor de serviços (PETERS, 2000). Mas é muito importante identificar os factores que levam a que essa estimativa seja muito elevada entre consumidores valiosos (FALTYS, 2000). Existe até o termo *Churn*, derivado de uma palavra de origem inglesa (que significa agitação ou movimento) usado para designar a falta de "fidelidade" dos clientes em determinados negócios. A taxa de *Churn* é definida pela seguinte fórmula:

clientes perdidos no mês / (clientes existentes no mês + clientes adicionados no mês).

Para alcançar o sucesso num mercado altamente competitivo é necessário compreender e tratar os clientes, e potencias clientes, como indivíduos, ou seja, para conquistar uma vantagem competitiva, o serviço e o marketing devem ser individualizados. Até há bem pouco tempo atrás, o importante era "conquistar o cliente a qualquer preço". A cultura actual baseia-se numa metodologia um pouco diferente: "reter o cliente" e "conquistar o bom cliente". É importante manter o cliente e evitar o *Churn*, pelo que o *Data Warehousing* é muito importante neste sentido.

De um modo geral, uma solução de DW tem por finalidade atender às necessidades de análise de informações dos utilizadores, tais como gerir e comparar as operações actuais com as passadas e prever situações futuras. Diversos outros objectivos estão relacionados com a implementação de um DW, tais como:

- Transformar, consolidar e racionalizar as informações dispersas por diversas bases de dados e plataformas, permitindo análises estratégicas bastante eficazes em informações antes inacessíveis ou subaproveitadas;
- Tornar as informações corporativas acessíveis para o seu entendimento e uso;
- Oferecer os fundamentos e os recursos necessários para um sistema de apoio à decisão eficiente, fornecendo dados integrados e históricos que servem desde a alta direcção, que necessita de informações mais resumidas, até às camadas de nível mais baixo, onde os dados detalhados ajudam a observar os aspectos mais tácticos de uma empresa.

O desafio dos sistemas de *Data Warehouse* é transformar os dados operacionais, distribuídos heterogeneamente pela organização, em informação estratégica agregada para suporte a processos de tomada de decisão, que garantam a competitividade exigida pelo negócio. Para este objectivo, a construção de um *Data Warehouse* está apoiada por um processo sistemático

que compreende algoritmos, ferramentas, técnicas e uma arquitectura especialmente concebida para facilitar o armazenamento de grandes volumes de dados. Isto para permitir a obtenção de informação estratégica a partir da execução de consultas complexas ao repositório, dentro de aspectos restritos de tempo de resposta.

As bases de dados são de grande importância para uma empresa. Sempre foi difícil analisar os dados nelas existentes porque geralmente as grandes empresas detêm um volume imenso de dados espalhados em diversos sistemas. Conseguir informações que permitissem a tomada de decisões baseadas num histórico de dados tornava-se uma missão bastante difícil sendo necessário muitos recursos. O *Data Warehouse* auxilia no processo de tomada de decisões com o objectivo de manter os clientes e melhorar os serviços oferecidos.

Quando as organizações investem no desenho do seu *Data Warehouse* investem simultaneamente na consolidação dos seus conceitos de negócio, na melhoria da qualidade dos seus dados e numa reestruturação dos seus processos de trabalho. O objectivo deste sistema é ter disponível informação actualizada para servir o negócio.

Alguns objectivos importantes de um *Data Warehouse*, passam por trazer melhores produtos para o mercado em tempo mais oportuno, bem como analisar diariamente informação de vendas e tomar decisões rapidamente que podem afectar significativamente o bom desempenho de uma empresa. O *Data Warehousing* pode ser efectivamente a solução para a manutenção do lado competitivo de uma empresa.

5 O Processo de Construção e Manutenção Evolutiva de um Data Warehouse

O primeiro passo no processo de construção de um *Data Warehouse* é definir:

- O que se quer analisar (factos/atributos);
- Sobre que perspectivas (dimensões).

A título de exemplo, numa empresa de telecomunicações pode pretender analisar-se o número de chamadas sob várias perspectivas (tempo (ano/mês/dia), tipo de cliente (empresarial/particular) e tipo de produto (pré-pago/pós-pago/plano controlado de custos).

É fundamental observar e analisar o comportamento da empresa, onde o papel dos gestores e analistas é essencial. A interacção deve ser a mais pormenorizada possível, de modo a que o teor dos dados seja o mais transparente possível e orientado para o âmbito em causa.

Posteriormente deverá ocorrer um envolvimento e participação da equipa de TI (Tecnologias de Informação). Nesta fase deve existir uma interacção entre a equipa de TI e os gestores e analistas, que possuem o conhecimento e o entendimento sobre o negócio.

A fase seguinte baseia-se na identificação de indicadores das áreas estratégicas, bem como no planeamento da execução da resolução do âmbito em causa, e validação dos indicadores que foram seleccionados. Devem ser definidos os indicadores e os critérios utilizados, bem como deve ser planeado um conjunto de factores para a implementação: recursos, orçamentos, facilidades, dificuldades, etc.

A próxima etapa é a identificação das fontes de dados. Os dados em "bruto" e os *inputs* para o DW devem ser identificados (transacções, dados externos, etc.)

Para a implementação do DW o gestor deve analisar de uma forma contínua eventuais alterações, políticas e procedimentos que poderão ter impactos nos *inputs* e *outputs* do DW.

Algumas etapas importantes que devem ser consideradas na construção do *Data Warehouse* são as seguintes:

- Escolher quais os processos que se pretendem modelar, construindo uma tabela de Factos correspondendo a cada processo que for escolhido. Pode definir-se uma tabela de Factos, por uma tabela que está interligada a um conjunto de tabelas de Dimensão. Estas contêm a descrição dos factos (dados) armazenados na tabela de Factos.
- Definir a granularidade de cada tabela de factos para cada processo, especificando qual o nível de detalhe a ser representado pelos factos.
- Definir as dimensões de cada tabela de factos. As dimensões serão definidas conforme a necessidade do utilizador. É necessário identificar os cruzamentos de dados que interessam. Por exemplo, numa empresa de telecomunicações pode ser necessário obter informações do cartão por campanhas, serviços, estado do cartão, etc.
- Especificar os factos.
- Analisar os atributos das dimensões, de modo a estabelecer descrições completas e terminologias apropriadas.
- Decisões sobre o projecto: agregações, dimensões heterogéneas e mini-dimensões. Soluções de indexação especial podem ser vantajosas, tais como, o uso de índices *bitmap* que devem ser utilizados em colunas com baixa cardinalidade. Por exemplo, o endereço na dimensão cliente não necessita de ser indexado, mas por outro lado campos como idade e sexo que são amplamente utilizados devem ser indexados. Também podem ser criadas mini-dimensões para melhorar a performance. Existem atributos que são muito utilizados nos cruzamentos e a performance dos movimentos através destes atributos cai significativamente.
- Preparar as dimensões para suportar evoluções.
- Definir o espectro temporal da base de dados.
- Definir a frequência com que os dados devem ser extraídos e carregados no *Data Warehouse*.

Um *Data Warehouse* implica um contínuo desenvolvimento evolutivo a par de uma eficiente manutenção. A figura seguinte ilustra várias fases possíveis num processo de manutenção evolutiva de um DW:

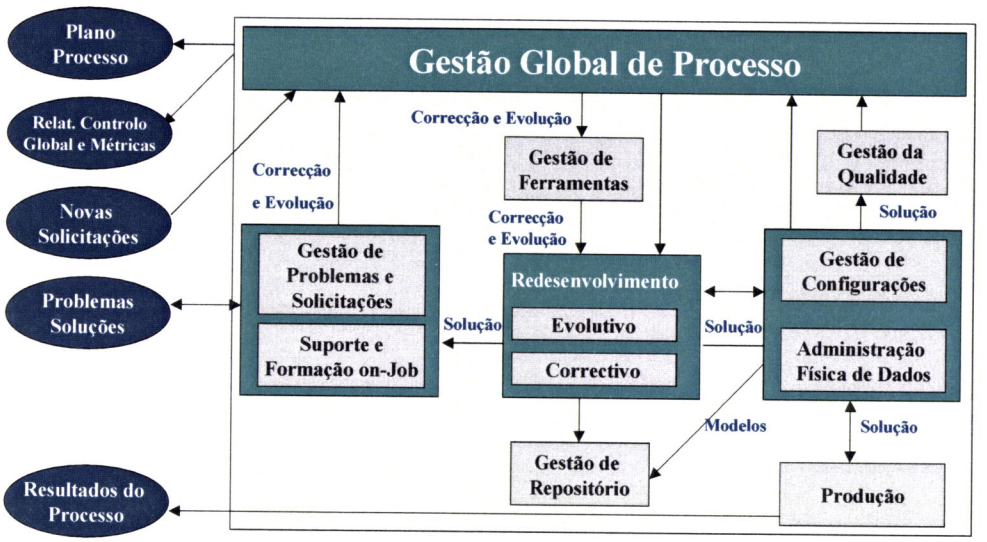

Figura 17 – Processo de Data Warehouse

- **Plano de Processo**: Definição de estratégias adequadas, que permitam atingir objectivos concretos. Visão global dos objectivos do *Data Warehouse* (actuais e futuros).
- **Relatórios de Controlo Global e Métricas:** Administração do *Data Warehouse* (tempos de carregamento, crescimento de espaço em disco, controlo de erros de carregamento, etc).
- **Novas Solicitações:** Pedidos de disponibilização de novos modelos no *Data Warehouse* por parte das direcções cliente, para que estejam disponíveis novas informações. Pedidos de alteração às estruturas das tabelas, optimizações aos carregamentos, etc.
- **Problemas e Soluções:** Análise a problemas detectados no *Data Warehouse* e análise de possíveis soluções para esses problemas.
- **Gestão de Ferramentas:** Actualizações a ferramentas de controlo de versões, ferramentas de desenvolvimento, ferramentas de ETL, ferramentas de *front-end* e manuais.
- **Gestão de Problemas e Solicitações:** Utilização de aplicações (de intranet por exemplo) para gerir as solicitações efectuadas por equipas

de testes e desenvolvimento (replicação de tabelas do ambiente de produção para outros ambientes, pedidos de aumento de espaço em *tablespaces* de dados e índices, *export* de ficheiros, etc.)

- **Suporte e Formação On-Job:** Serviço de *helpdesk* DW e formação para os utilizadores do *Data Warehouse.*
- **Re-desenvolvimento (Evolutivo/Correctivo):** Correcções que sejam necessárias ou optimizações a desenvolvimentos. Um projecto de *Data Warehouse* nunca pode ser dado como concluído dada a necessidade de se adaptar continuamente às necessidades da empresa e à evolução do seu modelo de negócio. Só através desta evolução contínua se pode trazer os benefícios previstos e reconhecidos como fundamentais à melhoria da performance empresarial.
- **Gestão de Repositório:** A gestão do repositório é feita pelo DBA. Um *Data Base Administrator* é a pessoa responsável pela administração e gestão de uma base de dados. Efectua operações diversas, tais como, a instalação de componentes, cálculo e dimensionamento de espaços para objectos, criação de utilizadores, fornecimento de permissões, análises periódicas às bases de dados considerando aspectos como estimativas de crescimento de espaço em tabelas, execução de *backups* de segurança, etc.
- **Gestão da Qualidade:** A gestão da qualidade pode ser garantida através da aplicação de metodologias de qualidade e testes por equipas especializadas nesta área.
- **Gestão de Configurações/Administração Física de dados:** Actualizações e alterações na metadata, que faz a indicação da descrição dos dados, do seu ambiente, como são manipulados e para onde são distribuídos.
- **Produção:** Trata-se do ambiente final dos desenvolvimentos. É neste ambiente que são executados diariamente, semanalmente ou mensalmente diversos processos de carregamento de tabelas.
- **Resultados do Processo:** Os resultados obtidos em produção serão posteriormente analisados. São muitas vezes gerados relatórios feitos

em *MicroStrategy* por exemplo, os quais permitem fazer análises aos dados guardados nos repositórios, de modo a auxiliar em decisões de negócio. O *Data Mining* (processo analítico para explorar grandes quantidades de dados) e o *Reporting* (ferramentas que permitem analisar informação de forma simples, abrangente e intuitiva através de relatórios dinâmicos, interactivos e personalizáveis) são outros componentes essenciais, já que fornecem as ferramentas que permitem explorar e analisar os dados guardados no *Data* Warehouse e nos *Data Marts*. Os dados são utilizados em análises efectuadas de acordo com as regras orientadas às entidades do negócio e perseguem objectivos de cariz comercial, como por exemplo:

- Segmentação de clientes;
- Escolha de alvos de campanhas;
- Aferição da rentabilidade de clientes e produtos;
- Medição do sucesso de campanhas e lançamento de novos produtos;
- Pesquisas de mercado, etc.

6 Abordagens e Metodologias de Qualidade e Testes de um Data Warehouse

6.1 ENQUADRAMENTO DE UMA EQUIPA DE TESTES

Uma equipa de testes funciona numa perspectiva de disponibilização de *feedbacks*. A equipa recepciona o projecto a testar e deverá enviar um *feedback*, funcionando como uma função matemática em que é recebido um conjunto de *inputs* e é devolvido um resultado, que neste caso pode ser considerado como *Booleano* ("Aprovado" ou "Não Aprovado").

Os *feedbacks* fornecidos devem ser fundamentados com base em testes e validações que deverão garantir tanto ao gestor de projecto, como à equipa de sistemas de produção, um funcionamento robusto e fiável. Aquando da conclusão do desenvolvimento, deverá ser entregue o projecto à equipa de testes e após um processo de validação e testes, será desenvolvido o relatório de testes. Este deverá descrever o processo de validação e testes aplicados, as recomendações, e o veredicto final sobre a capacidade do projecto reunir, ou não, as condições necessárias para avançar para a próxima fase (entrada no ambiente de produção).

A equipa de testes deverá tentar antecipar todo e qualquer tipo de problemas e tentar que as equipas de desenvolvimento sigam determinadas regras de programação que vêem ao encontro das necessidades dos ambientes finais. É portanto objectivo da equipa de testes incentivar e divulgar as melhores práticas, técnicas e metodologias de desenvolvimento junto das equipas de desenvolvimento.

O envolvimento da equipa de testes nos projectos deverá ocorrer na fase final da análise, entre a fase de conclusão do documento de análise técnica (documento que contém todas as especificações técnicas do projecto), e o início dos desenvolvimentos. Este envolvimento na fase inicial ajudará a perceber a abrangência e profundidade dos projectos, permitindo antever alguma da complexidade associada e preparar antecipadamente condições que possibilitem os melhores testes e validações.

6.2 OBJECTIVOS DA APLICAÇÃO DE METODOLOGIAS DE QUALIDADE E TESTES

Os objectivos da aplicação de metodologias de qualidade e testes caracterizam-se por:

- Testar e validar os desenvolvimentos efectuados:
  - Validar a correcta nomenclatura de ficheiros (itens dos projectos) de acordo com manual de desenvolvimento da organização;
  - Validar a documentação desenvolvida, garantindo que é correctamente orientada aos fins a que se propõe;
  - Validar a correcta distribuição de itens em aplicações de controlo de versões;
- Validar impactos intra-sistemas;
- Identificar erros no processo de ETL;
- Identificar problemas na performance dos projectos e nos dados que são disponibilizados. Verificar que o conteúdo de dados de cada tabela está em conformidade com o esperado, de acordo com as fontes utilizadas e as regras de mapeamento definidas;
- Garantir boas práticas de programação;
- Garantir estabilidade e performance no funcionamento de aplicações;
- Garantir que todo o *software* é funcional e que executa com sucesso as funções para o qual foi concebido.

6.3 CICLO DE DESENVOLVIMENTO

O ciclo de vida de um desenvolvimento é garantido através dos vários estados pelos quais passam os projectos. A figura abaixo ilustra o ciclo vida de desenvolvimento de um novo projecto para o *Data Warehouse*:

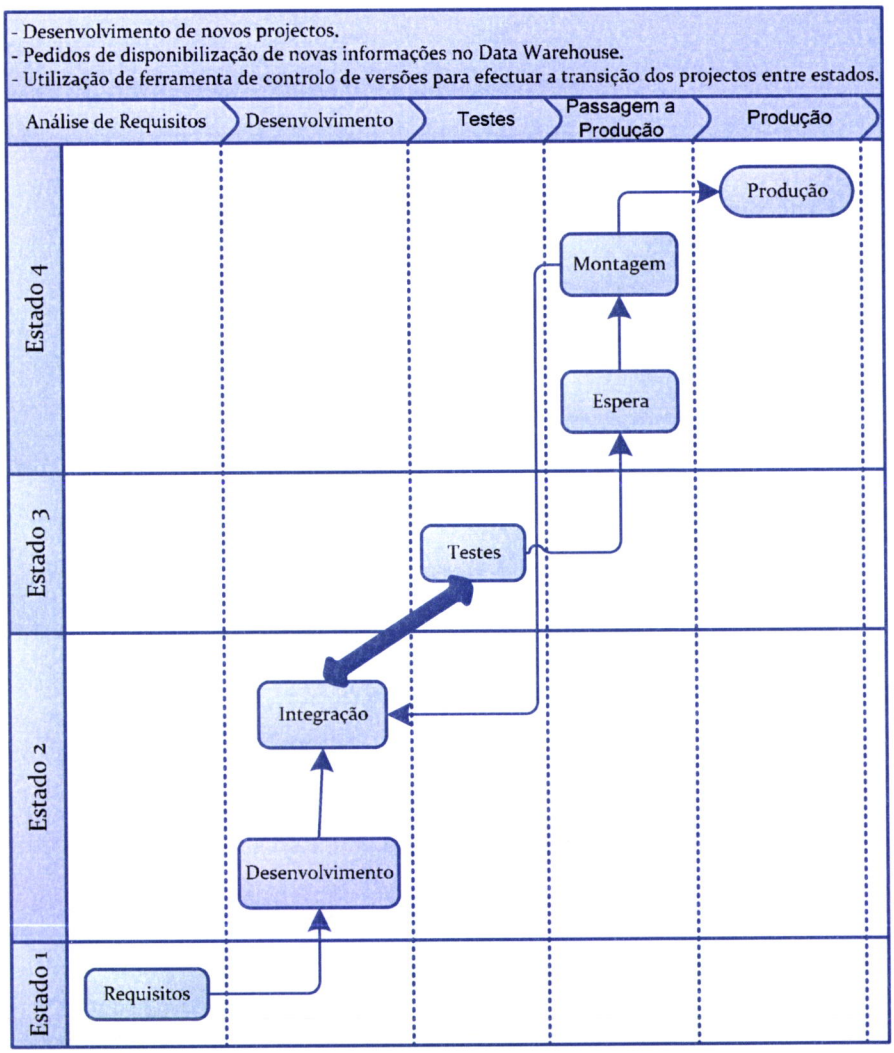

Figura 18 – Ciclo de desenvolvimento para novos projectos

- **Requisitos:** É nesta fase que são elaborados os documentos de especificação técnica e funcional por parte das equipas de desenvolvimento.
- **Desenvolvimento:** Desenvolvimento de projectos por parte das equipas de desenvolvimento.

- **Integração:** Um projecto que se encontre neste estado pode significar duas situações. Trata-se da versão final de um desenvolvimento e ainda não foi submetido a testes, ou então já foi submetido a testes e o mesmo não reunia condições para que fosse promovido para o nível imediatamente acima. As iterações entre a equipa de testes e as equipas de desenvolvimento são efectuadas entre o estado testes e o estado integração.
- **Testes:** É neste estado que são efectuados os testes aos desenvolvimentos. Deve ser garantido um ambiente de testes o mais real possível e próximo ao ambiente de produção. Para isso, são utilizadas máquinas, utilizadores e bases de dados no ambiente de testes que "simulem" as máquinas, os utilizadores e as bases de dados de produção.
- **Espera:** Após aprovação do projecto pela equipa de testes e validação efectuada pelo gestor de desenvolvimento, o projecto seguirá para este estado ficando a aguardar a passagem a produção. Por vezes existem projectos que apenas podem seguir para produção numa determinada data por algum motivo específico. Projectos prioritários poderão entrar primeiro.
- **Montagem:** É neste estado que é efectuada a instalação do projecto no ambiente de produção. Por vezes, o projecto pode retornar ao estado de integração caso inesperadamente surja algum problema.
- **Produção:** Representa o estado final de um projecto.

No entanto, mesmo após a entrada em produção podem surgir situações de emergência que necessitam de ser corrigidas. Os desenvolvimentos que entram para correcções de emergência seguem para produção sem passar por testes, visto tratarem-se de correcções pontuais e específicas a nível dos itens (ficheiros) dos projectos. A figura que se segue pretende mostrar o ciclo de desenvolvimento para correcções de emergência:

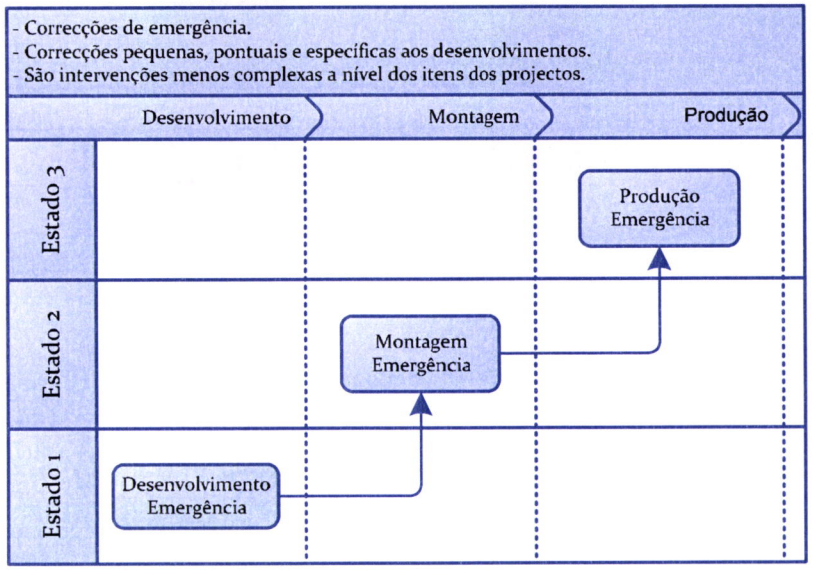

Figura 19 – Correcções de emergência

- **Desenvolvimento Emergência:** Estado onde são efectuadas correcções de emergência aos itens dos projectos por parte das equipas de desenvolvimento.
- **Montagem Emergência:** É nesta fase que é efectuada a instalação das alterações de emergência.
- **Produção Emergência:** Quando o projecto atinge este estado, é necessário fazer o *merge* dos seus itens com os itens do projecto que se encontra no ambiente de produção.

No entanto, por vezes surge a necessidade de se efectuar intervenções aos projectos um pouco mais complexas, dada a necessidade de manutenção correctiva e evolutiva do *Data Warehouse*. Estas podem representar um elevado impacto em toda a actividade de testes e validações. Por defeito, todos os projectos que se encontrem nesta situação devem ser testados tal e qual os outros projectos. Alguns projectos têm associadas situações de urgência, tendo naturalmente de ser tratados de forma privilegiada. A figura abaixo pretende ilustrar esse ciclo de correcção e manutenção evolutiva:

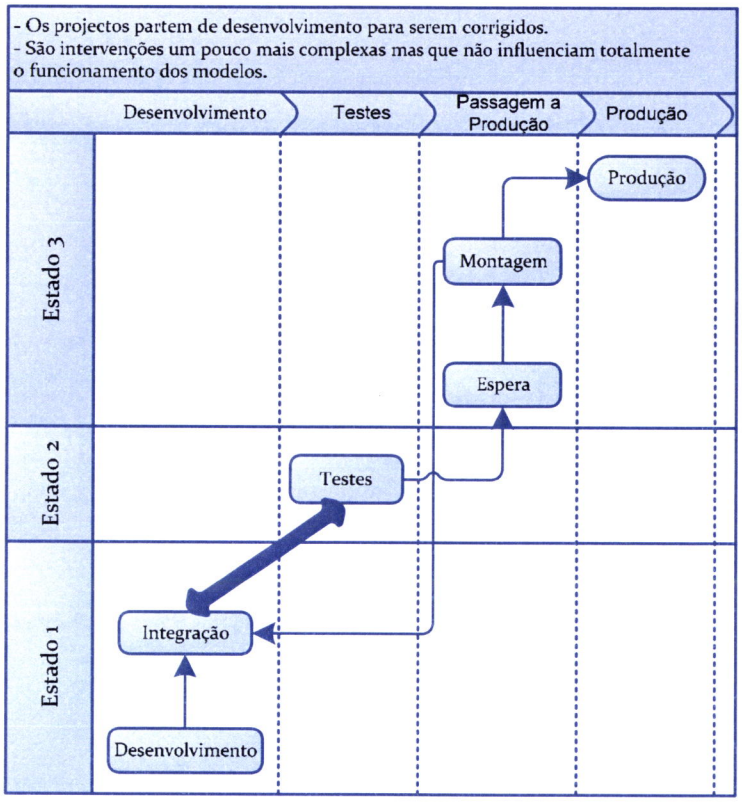

Figura 20 – Ciclo de correcção e manutenção evolutiva

6.4 OPERACIONALIZAÇÃO DE UMA EQUIPA DE TESTES

6.4.1 FLUXO DE INTERACÇÃO

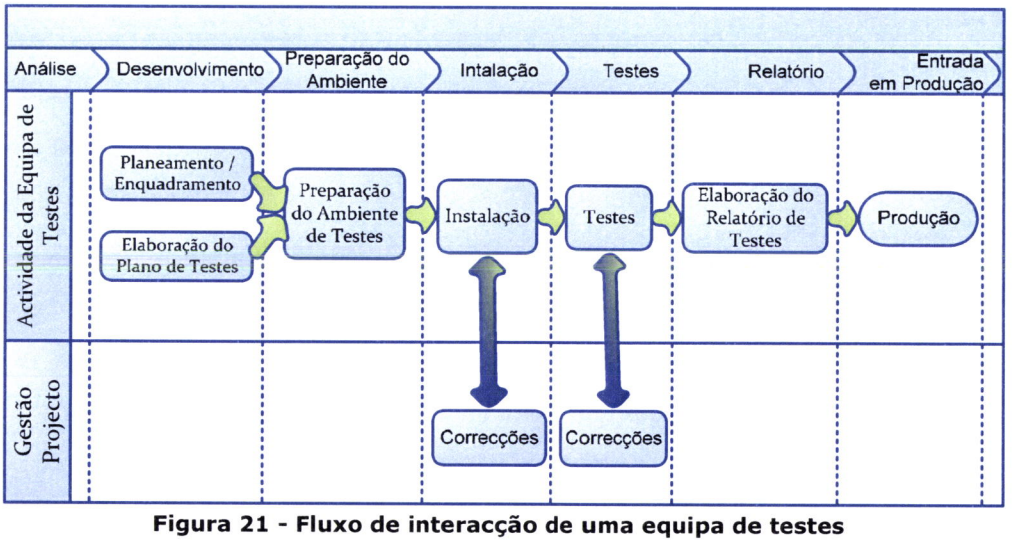

Figura 21 - Fluxo de interacção de uma equipa de testes

6.4.1.1 PLANEAMENTO/ENQUADRAMENTO

Esta fase caracteriza-se pelo agendamento e planeamento do trabalho a realizar sobre o desenvolvimento em curso. O gestor de projecto dá a conhecer o plano de trabalhos e a respectiva estratégia à equipa de testes. Após a conclusão do documento de análise técnica é feita uma apresentação das tarefas que vão ser realizadas, onde devem ser planeados e identificados os seguintes pontos:

- Objectivo do projecto;
- Esclarecimento de dúvidas relativas à especificação técnica do projecto;
- Data de inicio e de fim prevista para o projecto;
- Data de inicio e de fim para a fase de testes e validações;
- Data de entrada em produção;
- Objectos envolvidos;
- Impacto em outros sistemas;
- Documentação a elaborar;
- Tipo de testes a efectuar: carga, desempenho, etc.

Com base nas definições apresentadas e na disponibilização da análise técnica, a equipa de testes elaborará o plano de testes. O responsável de projecto deverá solicitar à equipa de desenvolvimento o maior detalhe possível na informação exposta no documento de especificação técnica, para que seja possível a construção de um plano de testes fiável. Deste plano de teste devem constar:

- Estratégias de teste;
- Ambiente dos testes;
- Casos de teste. Um caso de teste deve conter:
    - **Descrição:** descrição do caso de teste que vai ser efectuado;
    - **Pré-requisitos:** dependências existentes para a execução do caso de teste;
    - **Dados de teste:** breve descrição dos itens, objectos e funcionalidades a testar;

- **Resultados esperados:** Especificação de quais os resultados esperados, ou seja, o *output* gerado depois de executada determinada acção;
- **Acções de execução:** Especificação de quais as acções a executar no caso de teste.

No entanto, o plano de testes poderá vir a ser alterado no final dos desenvolvimentos, para que seja incluída informação mais pormenorizada sobre os casos de teste.

6.4.1.2 Preparação do Ambiente de Testes

Após o agendamento de datas e elaboração do programa de trabalho, deve ser iniciada a preparação do ambiente de testes. Durante esta fase, deverá ser definido como e quando serão os ambientes replicados. Esta fase deverá avançar assim que se tenha um conhecimento suficiente do projecto, para que se possa solicitar às equipas de sistemas de produção a replicação dos objectos necessários aos testes. Devem ser efectuadas todas as acções necessárias à preparação do ambiente de testes (cópia de tabelas, acessos a máquinas, etc). Pretende-se que o ambiente de testes seja o mais semelhante possível ao ambiente de produção, tanto a nível de *filesystem* como a nível de bases de dados. Deve ser efectuada uma preparação antecipada do ambiente de testes de forma a minimizar os tempos de teste após a conclusão dos desenvolvimentos. Qualquer alteração relevante no projecto deverá ser comunicada de imediato à equipa de testes. Em suma, a fase de preparação do ambiente de testes caracteriza-se pela calendarização e identificação dos objectos a migrar para o ambiente de testes.

6.4.1.3 Fase de Instalação

Esta fase tem como objectivo a execução e respectiva validação da *checklist*. Este documento deverá conter a referência a todos os objectos envolvidos, e a explicação de como a instalação do projecto deverá ser efectuada. É o guia de

instalação do projecto tanto em pré-produção como em produção. Na data acordada, deverá ser entregue o projecto desenvolvido para que se inicie a sua instalação nos ambientes de testes.

Durante o decorrer de todas as actividades de instalação dos objectos é efectuada a avaliação e registo de todos os problemas que possam surgir. Durante esta fase, e sempre que surgirem problemas que ponham em causa o bom funcionamento do projecto, a equipa de desenvolvimento deverá ser alertada de modo a esclarecer o que deve ser efectuado para resolver a situação.

6.4.1.4 TESTES E OPERACIONALIZAÇÃO

Esta fase centra-se principalmente na implementação do plano de testes e na validação das regras e métodos aplicados no desenvolvimento. Os testes deverão ser efectuados em condições que recriem o melhor possível as condições reais do ambiente de produção. A equipa de testes deverá focalizar-se nos testes de carga, desempenho e funcionais. Os testes de carga pretendem avaliar o comportamento dos sistemas em situações de utilização intensiva, bem como as capacidades de escalabilidade. Têm também como objectivo avaliar a concorrência entre processos. Os testes de desempenho analisam o tempo de reposta do sistema e verificam o nível de utilização dos recursos. Os testes funcionais baseiam-se em garantir a correcta implementação dos requisitos funcionais.

6.4.1.5 *FEEDBACK* E RELATÓRIOS

A intervenção da equipa de testes com os responsáveis de desenvolvimento é efectuada através do envio de pareceres com um resumo dos problemas detectados e as recomendações (e-mails por exemplo). O responsável de desenvolvimento na posse dessa informação, tomará a decisão de proceder ou não ao *demote* do projecto para que sejam efectuadas as devidas correcções. Se um projecto, após ser testado em ambiente de testes (pré-produção) não apresentar erros, então o mesmo poderá ser promovido para o nível

imediatamente acima (*promote*). Caso um projecto apresente problemas a nível técnico ou funcional, o mesmo terá de sofrer um *demote* para que a correcção seja efectuada. Deste modo o projecto retorna ao estado imediatamente abaixo. Após um *demote* para correcção seguido de *promote* para o ambiente de testes, são de novo executados os casos de teste necessários para cumprir os critérios de aceitação.

A intervenção da equipa de testes deverá terminar com a elaboração de um relatório de testes. Este é composto por um sumário executivo, contendo um resumo dos principais pontos identificados durante os testes e de cada uma das fases do trabalho desenvolvido, e ainda as respectivas recomendações. Deverá conter também todos os relatórios compilados durante os testes. A tarefa da equipa de testes fica concluída aquando da aprovação do relatório pelo responsável de projecto.

O relatório de testes deverá conter:

- Objectivo e âmbito dos testes;
- Caracterização dos testes e validações;
- Plano de trabalho;
- Resultados dos testes;
- Recomendações e conclusões;
- Decisão sobre a aceitação;

Caberá ao responsável de projecto a decisão de permitir ou não que o projecto entre em produção, em função da informação que lhe foi disponibilizada. Após esta aprovação, a equipa de testes deverá efectuar a passagem do projecto para o nível acima.

6.5 O CONCEITO DE TESTE

O conceito de teste pode ser compreendido através de uma visão intuitiva ou até mesmo formal. Testar significa verificar através de uma execução controlada se o comportamento do sistema corre conforme aquilo que seria esperado. O objectivo principal é detectar erros, mostrando a quem desenvolve

se os resultados estão, ou não, de acordo com os padrões que foram estabelecidos. (Wikipedia)

6.6 Tipos de Testes

6.6.1 Testes de Sistema

Pretendem verificar o sistema do ponto de vista do utilizador não considerando portanto a estrutura interna ou a forma de implementação do sistema. São considerados por isso testes de caixa-preta, em que o componente de *software* a ser testado é abordado como se fosse uma caixa-preta não sendo considerado neste caso o comportamento interno do sistema. São testes globais em que todos os componentes do sistema são integrados. A abordagem de integração é não incremental, combinando-se antecipadamente todas as aplicações. São testes dinâmicos e por isso seguem o modelo tradicional de teste de programa, no qual o programa é executado com alguns casos de teste sendo os resultados analisados para verificar se o resultado obtido é o que seria expectável. Por vezes, o componente a ser testado pode ser uma função interna, uma funcionalidade, um conjunto de componentes, etc. Quem elabora o plano de testes de sistema não deverá exercer funções de programador no projecto.

6.6.2 Testes de Carga

Têm como objectivo avaliar o comportamento dos sistemas em situações de utilização intensiva e aferir capacidades de escalabilidade. Pretendem também avaliar a concorrência entre processos.

6.6.3 Testes de Desempenho

Analisar o tempo de resposta do sistema (tempos de execução dos processos) no ambiente de testes e garantir que o resultado obtido é o esperado. Apesar de se pretender um ambiente de testes o mais semelhante possível ao

ambiente de produção, o ambiente de testes nem sempre reflecte exactamente a realidade do ambiente de produção devido aos recursos das máquinas, às velocidades dos discos e a processos que se encontram a "correr" no momento da entrada em produção dos projectos.

6.6.4 Testes de Usabilidade

A usabilidade é uma característica daquilo que é utilizável e funcional. Estes testes têm por objectivo verificar a adequabilidade das interfaces homem/máquina.

6.6.5 Testes Funcionais

Determinar a correcta implementação dos requisitos funcionais. Por vezes a equipa de testes não tem conhecimento de todas as particularidades solicitadas pelo cliente. Com base na documentação que é disponibilizada e no conhecimento adquirido ao longo do projecto, a equipa de testes realiza os testes que entende serem adequados. Após a disponibilização de dados ao gestor de projecto, este poderá solicitar novos testes caso os que tenham sido efectuados não sejam suficientes para garantir a correcta validação da informação a ser disponibilizada.

6.6.6 Testes Unitários

Este tipo de testes corresponde a testes isolados aos componentes (serviços/funções). Tal como o nome indica, são testes a algumas partes ou unidades do sistema. São testes de caixa-branca que avaliam o comportamento interno de um componente de *software*. É necessário ter acesso ao código fonte de todos os componentes disponíveis, de modo a controlar o que foi e o que não foi testado. São testes dinâmicos que requerem que o processo seja executado, no qual são executados alguns casos de teste e os resultados da execução são examinados para verificar se o processo operou como seria esperado. Este tipo de testes é efectuado pelas equipas de desenvolvimento

após a fase de desenvolvimento do projecto. Quem elabora o plano de testes unitários não deverá exercer funções de programador no projecto.

6.6.7 Testes de Integração

Pretendem avaliar se as interfaces entre componentes duma mesma aplicação interagem conforme o especificado. A alteração num determinado processo que tenha impactos em mais do que um modelo, nem sempre obriga a que todos os modelos sejam testados. Esta situação poderia implicar tempos de teste bastante elevados, que não podem ser suportados dada a urgência de entrada em produção dos projectos. Seguem uma abordagem incremental, em que o programa é testado em pequenos segmentos. Esta abordagem permite uma correcção de erros mais fácil e a interface tem maior probabilidade de ser testada completamente. Começa por se integrar e testar módulos de nível inferior e vai-se subindo na hierarquia até que o módulo final seja testado.

6.7 Exemplos de Testes

6.7.1 Validação de itens

A finalidade deste teste é verificar se os itens relacionados com o projecto estão de acordo com as normas do manual de desenvolvimento da organização.

6.7.2 Distribuição dos itens e preparação do ambiente de trabalho

Pretende-se com este teste verificar a correcta descrição sobre a colocação do processo em produção. Deve ser possível efectuar a correcta distribuição e criação de todos os pontos definidos na *checklist*. Deve ser garantido um ambiente de trabalho o mais próximo possível do existente em produção.

6.7.3 EXECUÇÃO DO PROCESSO E VERIFICAÇÃO DOS LOGS

Pretende-se garantir a execução do processo implementado e o correcto *output* para os ficheiros de log. É também objectivo deste teste garantir que não permanecem objectos no *filesystem* ou na base de dados, que sejam desnecessários.

6.7.4 COMPARAÇÃO ENTRE PROCESSO ANTIGO / PROCESSO NOVO

É objectivo deste teste verificar quais as diferenças que são obtidas aquando do carregamento de tabelas pelo processo antigo e pelo processo novo. Pode fazer sentido que se obtenham diferenças quando o motivo da intervenção foi adicionar uma nova fonte para carregamento de um novo produto/serviço, ou por exemplo, quando foi alterado um processo para passar a incluir determinado tipo de registos que anteriormente estava a ser descartado. No entanto, por vezes são efectuadas optimizações aos processos. Neste caso o objectivo já será o de não obter diferenças.

Exemplo:

Executar o processo antigo e guardar um backup das tabelas:

```
CREATE TABLE FACT_PRODUTO_OLD TABLESPACE TBS_TESTES AS
SELECT * FROM FACT_PRODUTO;
CREATE TABLE FACT_SERVICO_OLD TABLESPACE TBS_TESTES AS
SELECT * FROM FACT_SERVICO;
```

Correr o processo novo e executar as seguintes *queries* de comparação:

```
SELECT COUNT(*) FROM(
SELECT * FROM FACT_PRODUTO
WHERE COD_DIA=20080322
MINUS
SELECT * FROM FACT_PRODUTO_OLD
WHERE COD_DIA=20080322);
--Result: 0 rows

SELECT COUNT(*) FROM(
SELECT * FROM FACT_PRODUTO_OLD
WHERE COD_DIA=20080322
```

```
MINUS
SELECT * FROM FACT_PRODUTO
WHERE COD_DIA=20080322);
--Result: 0 rows


SELECT COUNT(*) FROM(
SELECT * FROM FACT_SERVICO
WHERE COD_DIA=20080322
MINUS
SELECT * FROM FACT_SERVICO_OLD
WHERE COD_DIA=20080322);
--Result: 0 rows

SELECT COUNT(*) FROM(
SELECT * FROM FACT_SERVICO_OLD
WHERE COD_DIA=20080322
MINUS
SELECT * FROM FACT_SERVICO
WHERE COD_DIA=20080322);
--Result: 0 rows
```

6.7.5 CONTAGEM DE REGISTOS DUPLICADOS

Pretende-se com este teste garantir a não existência de registos duplicados nas tabelas que são carregadas pelo processo. Dois registos são considerados distintos quando têm pelo menos uma coluna diferente. No entanto, são considerados registos duplicados, registos da mesma tabela que tenham a mesma chave primária.

Exemplo:

Execução de *queries* do tipo:

```
SELECT a.COD_SERVICO, a.COD_CONTA, a.COD_TAR_SERVICO,
a.COD_AGENTE, a.COD_ESTADO_SERV, a.COD_MOTIVO_DESACT_SERV,
a.COD_LOGIN, a.QTD_SERV, COUNT(*)
FROM FACT_SERVICO a
WHERE a.COD_DIA= 20080625 AND a.COD_SERV = 432
```

```
GROUP BY a.COD_SERVICO, a.COD_CONTA, a.COD_TAR_SERVICO,
a.COD_AGENTE, a.COD_ESTADO_SERV, a.COD_MOTIVO_DESACT_SERV,
a.COD_LOGIN, a.QTD_SERV
HAVING COUNT(*) > 1
--Result: 0 rows
```

6.7.6 Análise de integridade referencial

Verificação de que todas as ocorrências distintas dos vários campos que sejam atributos presentes nas tabelas actualizadas pelos processos em teste, constam nas respectivas tabelas de Referência.

Exemplo:

Execução de *queries* deste tipo, que devem retornar zero registos:

```
SELECT <COD_ATRIBUTO> FROM <FACT_*>
MINUS
SELECT <COD_ATRIBUTO> FROM <REF_ATRIBUTO>;

SELECT COUNT(*) FROM <FACT_*> a
WHERE NOT EXISTS
(SELECT * FROM <REF_ATRIBUTO> b
WHERE a.<COD_ATRIBUTO> = B.<COD_ATRIBUTO>)
```

6.7.7 Análise da coerência Factual – Agregada

Verificação de que todos os registos carregados nas tabelas Agregadas são provenientes das tabelas Factuais que sejam fontes dessas tabelas. Pretende-se verificar se existem registos nas tabelas Agregadas que não estejam presentes também nas tabelas Factuais.

Exemplo:

Execução de *queries* do tipo:

```
SELECT 200803 AS COD_MES, COD_TARIFARIO, DEF_PLAFOND, COUNT(*) AS
QTD_TAR_PLA
FROM FACT_PLAFOND PARTITION (FACT_PLAFOND_200803)
GROUP BY (COD_TARIFARIO, DEF_PLAFOND)
MINUS
```

```
SELECT COD_MES, COD_TARIFARIO, DEF_PLAFOND , QTD_TAR_PLA
FROM AGG_PLAFOND_M1
WHERE COD_MES = 200803;
--Result: 0 rows


SELECT COD_MES, COD_TARIFARIO, DEF_PLAFOND, QTD_TAR_PLA
FROM AGG_PLAFOND_M1
WHERE COD_MES = 200803
MINUS
SELECT 200803 AS COD_MES, COD_TARIFARIO, DEF_PLAFOND, COUNT(*) AS
QTD_TAR_PLA
FROM FACT_PLAFOND PARTITION (FACT_PLAFOND_200803)
GROUP BY (COD_TARIFARIO, DEF_PLAFOND);
--Result: 0 rows
```

6.7.8 TESTE DE REPROCESSAMENTO

Pretende-se com este teste garantir que, aquando de um reprocessamento do processo de carregamento das tabelas, os dados nelas contidas estão correctos, ou seja, que o número de registos existente nas tabelas usadas para o teste é o mesmo.

Exemplo:

Executar o processo de carregamento para o mês N.

Guardar backups das tabelas:

```
CREATE TABLE FACT_PRODUTO_RP TABLESPACE TBS_TESTES AS
SELECT * FROM FACT_PRODUTO;
CREATE TABLE FACT_SERVICO_RP TABLESPACE TBS_TESTES AS
SELECT * FROM FACT_SERVICO;
```

Executar novamente o processo para o mês N.

Comparar as tabelas, verificando se o número de registos é o mesmo:

```
SELECT * FROM FACT_PRODUTO_RP
MINUS
SELECT * FROM FACT_PRODUTO
--Result: 0 rows
```

```
SELECT * FROM FACT_PRODUTO
MINUS
SELECT * FROM FACT_PRODUTO_RP
--Result: 0 rows


SELECT * FROM FACT_TARIFARIO_RP
MINUS
SELECT * FROM FACT_ TARIFARIO
--Result: 0 rows

SELECT * FROM FACT_ TARIFARIO
MINUS
SELECT * FROM FACT_ TARIFARIO_RP
--Result: 0 rows
```

6.7.9 VALIDAÇÃO DO CORRECTO MAPEAMENTO DE NOVAS COLUNAS

No caso de projectos em que são adicionadas novas colunas às tabelas, é necessário verificar se o carregamento dessas colunas está a ser correctamente efectuado de acordo com os mapeamentos especificados no documento de análise técnica. São por isso executadas *queries* que simulem esses mapeamentos, sendo necessário confrontá-las com os dados que foram carregados para as tabelas.

6.7.10 COMPARAÇÃO/CONTABILIZAÇÃO DE REGISTOS ENTRE FONTES / TABELA CARREGADA

A finalidade deste teste é fazer uma comparação entre o número de registos existentes nas fontes e nas tabelas que são carregadas com base nessas fontes. Devem ser executadas *queries* de contagens sobre as fontes e sobre as tabelas que são carregadas pelo processo.

Exemplo:

Execução de *queries* para contagens nas fontes:

```
SELECT COUNT(*) FROM DSA_DESCONTO
WHERE TIPO_VALOR = 'CA' AND DESCONTO = 2244 AND COD_DIA=20080322
```

--Result: 239

SELECT COUNT(*) A FROM DSA_CAMPANHA WHERE CAMPANHA = 'EA08' AND OFERTA = 12 AND SERVICO = 4
--Result: 1933
Total de registos a carregar: 239+1933=2172

Execução de *queries* para contagens na tabela carregada:

SELECT COUNT(*) FROM(
*SELECT * FROM FACT_DESCONTO*
WHERE COD_DIA=20080322
MINUS
*SELECT * FROM FACT_DESCONTO_OLD*
WHERE COD_DIA=20080322)
--Result: 2172

6.7.11 IDENTIFICAÇÃO PONTUAL DE AMOSTRAS DE REGISTOS NAS FONTES E NAS TABELAS CARREGADAS

Este teste tem por objectivo a identificação de uma determinada amostra de registos nas fontes e na tabela que é carregada. Devem ser analisados ao pormenor esses registos, para que seja feita uma comparação fiável.

Exemplo:

Recolha aleatória de registos na tabela carregada:

*SELECT * FROM DSA_DUNNING SAMPLE (0.1)*

--Result:

*20080225|**000007511292**|8563762|N1|20080228|260,99|10285|856376253|260,99|10285|856376253|27,25|10285|856376244|009163008269|20070723|50,13|10285|856376267|009162519069|20070623|49,20|10285|856376255|000164609223|20061123|48,90|10285|856376248|000164170442|20061023|85,51|10285|856376226|000161715551|20060526|101245|00856376256*

Identificação da amostra de registos na fonte (por exemplo um ficheiro):

*/directoria/filesystem>grep **000007511292** 47_20080225.dat*

--Result:

*20080225|**000007511292**|000008563762|N1|20080228|260,99|10285|8563*
76253|260,99|10285|856376253|27,25|10285|856376244|009163008269|20
070723|50,13|10285|856376267|009162519069|20070623|49,20|10285|856
376255|000164609223|20061123|48,90|10285|856376248|000164170442|2
0061023|85,51|10285|856376226|000161715551|20060526|101245|008563
76256

6.7.12 VERIFICAÇÃO DA CORRECTA DISTRIBUIÇAO DE FICHEIROS FONTE

Existem processos que têm como fontes determinado tipo de ficheiros. O objectivo deste teste é verificar se após a execução do processo, os ficheiros se encontram correctamente distribuídos nas directorias dos respectivos modelos. Muitas das vezes os ficheiros são copiados de entre diversas máquinas remotas.

6.7.13 VERIFICAÇÃO DO CORRECTO CARREGAMENTO DE FICHEIROS

Da mesma forma que existem processos que fazem o carregamento de tabelas com base em determinados ficheiros, também existem processos que carregam ficheiros tendo por base determinados campos de tabelas. Este teste tem por finalidade verificar o correcto carregamento desses ficheiros. Neste caso, são efectuadas contagens aos registos do ficheiro e aos registos das tabelas fontes. São também analisados em detalhe esses registos (por amostra caso o número de registos existente no ficheiro seja muito elevado).

6.7.14 VALIDAÇÃO DE MÉTRICAS DAS TABELAS AGREGADAS

O objectivo desta validação é verificar o correcto cálculo (em termos de quantidade) de determinadas métricas das tabelas Agregadas, de acordo com as respectivas fontes.

Exemplo:

Execução de *queries* na fonte (tabela Factual):

```
SELECT SUM(a.QTD_UL_KB_GPRS) AS QTD_UL_KB_GPRS
SUM(a.QTD_DL_KB_GPRS) AS QTD_DL_KB_GPRS,
SUM(a.QTD_FACT_KB_GPRS) AS QTD_FACT_KB_GPRS
```

```
FROM FACT_GPRS PARTITION (FACT_GPRS_20080721) a, REF_DIA b,
PAR_PER c
WHERE a.COD_HORA = c.HORA AND a.COD_DIA = b.COD_DIA
AND a.COD_DIA BETWEEN c.DIA_INICIO AND c.DIA_FIM
AND b.COD_TIPO_DIA = c.COD_TIPO_DIA;

--Result:
QTD_UL_KB_GPRS  = 4757056
QTD_DL_KB_GPRS  = 26625021
QTD_FACT_KB_GPRS = 84291450
```

Execução de *queries* na tabela Agregada carregada pelo processo:

```
SELECT SUM(QTD_UL_KB_GPRS) AS QTD_UL_KB_GPRS,
SUM(QTD_DL_KB_GPRS) AS QTD_DL_KB_GPRS,
SUM(QTD_FACT_KB_GPRS) AS QTD_FACT_KB_GPRS
FROM AGG_GPRS_D1
WHERE COD_DIA = 20080721;

--Result:
QTD_UL_KB_GPRS = 4757056
QTD_DL_KB_GPRS = 26625021
QTD_FACT_KB_GPRS = 8429145
```

6.7.15 *Simulação do carregamento de registos* para as tabelas

Este teste tem por objectivo efectuar uma simulação do carregamento de registos para as tabelas, de acordo com as regras (mapeamentos) definidas no documento de análise técnica. Devem ser por isso executadas *queries* onde são aplicadas essas regras e confrontá-las com os dados carregados nas tabelas.

6.7.16 Verificação dos tempos de execução do processo

A finalidade deste teste é efectuar uma comparação do tempo de execução entre o processo antigo e o processo actualizado. É executado o processo antigo e registado o seu tempo de execução. Posteriormente é executado o processo novo. É espectável que exista uma melhoria significativa no tempo de execução.

6.7.17 VERIFICAÇÃO DA EXISTENCIA DE VALORES NULL EM TABELAS

Este teste pretende verificar a não existência de colunas com valores a null. Caso existam, deve ser encontrada, analisada e justificada essa situação.

Exemplo:

Execuções de *queries* do tipo:

*SELECT * FROM REF_CARD WHERE <ATRIBUTO> IS NULL*

6.7.18 COMPARAÇÃO DA ESTRUTURA DE TABELAS

Este teste baseia-se na comparação entre estruturas de tabelas, quando existe a necessidade de garantir que efectivamente duas tabelas possuem a mesma estrutura em *users* diferentes de uma base de dados. Neste caso podem ser executadas *queries* que efectuem essa comparação.

Por exemplo:

Execução de *queries* deste tipo, que devem retornar zero registos:

```
SELECT COLUMN_NAME, DATA_PRECISION, DATA_TYPE, DATA_LENGTH
FROM ALL_TAB_COLUMNS WHERE OWNER = 'USER1'
AND TABLE_NAME = 'REF_PRODUTO'
MINUS
SELECT COLUMN_NAME, DATA_PRECISION, DATA_TYPE, DATA_LENGTH
FROM ALL_TAB_COLUMNS WHERE OWNER = 'USER2'
AND TABLE_NAME = 'REF_PRODUTO';
--Result: 0 rows
```

6.8 VANTAGENS E DESVANTAGENS DA APLICAÇÃO DE METODOLOGIAS DE QUALIDADE E TESTES

As vantagens da aplicação de metodologias de qualidade e testes resumem-se nos seguintes aspectos:

- Redução significativa do risco de problemas sobre os sistemas de produção;
- Incremento da confiança nos sistemas;
- Redução do tempo de passagem a produção;
- Redução do número de reprocessamentos após passagens a produção;
- Optimização do funcionamento dos sistemas;
- Garantia de qualidade.

No entanto, podem considerar-se desvantagens os seguintes aspectos:

- Necessidade de um maior número de recursos (materiais e humanos), implicando gastos monetários mais elevados;
- O ambiente de testes nem sempre consegue reflectir exactamente a realidade do ambiente de produção, devido aos recursos das máquinas e a processos que se encontram a executar no momento da entrada em produção dos projectos. Esta situação pode levar por vezes a simulações de execução erradas.

7 Caso Prático

7.1 NOTA INTRODUTÓRIA

O caso prático abordado nesta dissertação, teve como base a aplicação de metodologias de qualidade e testes por parte de uma equipa de testes numa empresa de telecomunicações ("empresa Telco") da qual não foi permitido referir o nome. Deste modo, a informação aqui disponibilizada não é a real, pretende meramente ser utilizada a título de exemplo. O motivo desta decisão teve que ver com o facto do mercado da indústria das telecomunicações ser um mercado exponencialmente competitivo.

Ao conteúdo deste capítulo, foi dada maior relevância ao tema alvo desta dissertação (metodologias de qualidade e testes) do que propriamente aos requisitos funcionais e técnicos definidos pelas equipas de desenvolvimento.

7.2 APLICAÇÃO AO CASO PRÁTICO

Objectivo: O objectivo deste projecto é disponibilizar no *Data Warehouse* informação histórica sobre Leilões.

Requisitos/descrição: Informação para análise mensal de dados agregados de acordo com o tipo de cartão (pós-pago, pré-pago e plano controlado de custos):

- Descrição do nome do produto que esteve a leilão;
- Quantidade de trocas efectuadas no leilão;
- Número de pontos trocados;
- Quantidade de licitações;
- Número de pontos licitados;
- Licitação mínima;
- Para análise de perfil de clientes:
    - Informação por número de telemóvel;
    - Quantidade de pontos licitados por leilão;
    - Quantidade de pontos licitados no total de leilões em que o cliente participou;

- Quantidade de licitações efectuadas por leilão;
- Datas dos leilões em que o cliente participou (dia/mês/ano);
- Número de participações totais em diferentes leilões;
- Descrição das ofertas a leilão em que o cliente participou.

Conjunto de itens associado ao projecto:

Tipo de item	Nome do item
.doc	Analise_Funional_Leiloes.doc Analise_Tecnica_Leiloes.doc Manual_Operacao_Leiloes_DW.doc Checklist_Leiloes.doc
.dsx	LeiloesPontos.dsx
.sh	dsLeilaoPontosFactCrg.sh dsLeilaoPontosFactExt.sh dsLeilaoPontosFactPrc.sh dsLeilaoPontosRefCrg.sh dsLeilaoPontosRefExt.sh
.ddl	FACT_LEILAO_PONTOS.ddl REF_LEILAO_PONTOS_ESTADO.ddl REF_LEILAO_PONTOS.ddl REF_LEILAO_PONTOS_TIPO.ddl
.idx	FACT_LEILAO_PONTOS.idx REF_LEILAO_PONTOS_ESTADO.idx REF_LEILAO_PONTOS.idx REF_LEILAO_PONTOS_TIPO.idx

Tabela 4 – Itens contidos no projecto

Tabelas que serão disponibilizadas no *Data Warehouse* para responder ao requisito:

FACT_LEILAO_PONTOS:

- Tabela Factual com todas as licitações efectuadas;
- Refrescada e particionada mensalmente;
- Latência de 12 meses;

REF_LEILAO_PONTOS:

- Tabela de Referência com todos os leilões já terminados e processados;
- Refrescada mensalmente;

REF_LEILAO_PONTOS_ESTADO:

- Tabela de Referência com todos os estados de licitação possíveis;
- Refrescada mensalmente;

REF_LEILAO_PONTOS_TIPO:

- Tabela de Referência com os tipos de leilão;
- Refrescada mensalmente.

Após o desenvolvimento do projecto por parte da equipa de desenvolvimento e feito o *promote* do mesmo para o estado de Testes, foi necessário efectuar a preparação do ambiente de testes. Esta preparação impôs a correcta distribuição dos itens de acordo com o que é indicado na *checklist* tendo em conta as normas de desenvolvimento. Foi necessário garantir que o ambiente de pré-produção estava coerente com o existente em produção, tendo sido solicitado por isso o carregamento das tabelas fonte às equipas de produção. Visto que este desenvolvimento disponibiliza novas tabelas no *Data Warehouse* foi ainda necessário o pedido de criação dos *tablespaces* de dados e índices com espaço suficiente para o carregamento das tabelas (pedidos efectuados através de uma aplicação de intranet).

Foi também elaborado o plano de testes tendo por base o documento de especificação técnica. Em seguida é apresentado o essencial do referido plano de testes:

PLANO DE TESTES DE VERIFICAÇÃO

1. **Objectivo:** O presente documento descreve o conjunto de testes a aplicar ao desenvolvimento, pretendendo assim detectar eventuais erros e impactos negativos da sua entrada em produção.
2. **Ambiente dos testes:** máquina - ATLANTA, base de dados - TESTES.
3. **Casos de teste:**

TESTE 1 – Distribuição dos itens e preparação do ambiente de trabalho	
Descrição	Pretende-se com este teste verificar a correcta descrição sobre a colocação do processo em produção
Pré-requisitos	Deve-se garantir um ambiente de trabalho o mais próximo possível do existente em produção
Dados de teste	Checklist de entrada em produção
Resultados esperados	Deve ser possível criar e distribuir todos os pontos definidos pela checklist de entrada em produção
Acções de execução	Executar as acções definidas pela documentação

TESTE 2 – Execução do processo e verificação dos logs	
Descrição	Pretende-se garantir a execução do processo implementado e o correcto output para os ficheiros de log. Também se deverá garantir que não permanecem objectos no FS ou na BD que sejam desnecessários
Pré-requisitos	Correcta execução do teste 1
Dados de teste	Logs do processo
Resultados esperados	Espera-se que todas as operações corram com sucesso, que sejam descritos todos os passos nos ficheiros de log e que não permaneçam objectos nos sistemas que sejam desnecessários
Acções de execução	Execução dos scripts do processo

TESTE 3 – Verificação de dados da tabela FACT_LEILAO_PONTOS	
Descrição	Garantir o correcto preenchimento da tabela FACT_LEILAO_PONTOS de acordo com as suas fontes
Pré-requisitos	Correcta execução do teste 2
Dados de teste	TABELA: FACT_LEILAO_PONTOS; FONTES: TB_LICITACAO, TB_LEILAO
Resultados esperados	Espera-se que o carregamento da tabela FACT_LEILAO_PONTOS seja correctamente efectuado
Acções de execução	Fazer uma análise comparativa aos registos existentes nas fontes e na tabela carregada

TESTE 4 – Verificação de dados nas tabelas de Referência	
Descrição	Garantir o correcto preenchimento das tabelas REF_LEILAO_PONTOS, REF_LEILAO_PONTOS_ESTADO e REF_LEILAO_PONTOS_TIPO
Pré-requisitos	Correcta execução do teste 3
Dados de teste	TABELAS: REF_LEILAO_PONTOS, REF_LEILAO_PONTOS_ESTADO e REF_LEILAO_PONTOS_TIPO FONTES: TBLIC_ESTADO, TB_LEILAO e ficheiro REF_PONTOS_TIPO_LEILAO.txt
Resultados esperados	É espectável que as tabelas de Referência sejam correctamente carregadas
Acções de execução	Fazer uma análise comparativa aos registos existentes nas fontes e na tabela carregada

TESTE 5 – Integridade Referencial	
Descrição	Pretende-se garantir que todos os códigos COD_PONTOS_LEILAO, COD_PONTOS_ESTADO_LICITACAO, COD_CARD e COD_U_CARD existentes na tabela FACT_LEILAO_PONTOS estão presentes nas respectivas tabelas de Referência
Pré-requisitos	Correcta execução do teste 4
Dados de teste	FACT_LEILAO_PONTOS, REF_LEILAO_PONTOS, REF_LEILAO_PONTOS_ESTADO, REF_CARD, REF_UNIC_CARD
Resultados esperados	Espera-se que as queries usadas para teste retornem zero registos, demonstrando que todos os códigos existentes na tabela Factual estão presentes nas devidas tabelas de Referência
Acções de execução	Execução das seguintes queries: *SELECT COD_PONTOS_LEILAO FROM FACT_LEILAO_PONTOS* *MINUS* *SELECT COD_PONTOS_LEILAO FROM REF_LEILAO_PONTOS* *SELECT COD_PONTOS_ESTADO_LICITACAO FROM FACT_LEILAO_PONTOS* *MINUS* *SELECT COD_PONTOS_ESTADO_LICITACAO FROM REF_LEILAO_PONTOS_ESTADO* *SELECT COD_CARD FROM FACT_LEILAO_PONTOS* *MINUS* *SELECT COD_CARD FROM REF_CARD* *SELECT COD_U_CARD FROM FACT_LEILAO_PONTOS* *MINUS* *SELECT COD_U_CARD FROM REF_UNIC_CARD*

TESTE 6 – Contagem de duplicados após carregamento inicial	
Descrição	Pretende-se garantir a não existência de registos em duplicado na tabela FACT_LEILAO_PONTOS após um carregamento inicial
Pré-requisitos	Correcta execução do teste 5
Dados de teste	FACT_LEILAO_PONTOS_CI_04
Resultados esperados	Espera-se que a query usada para teste não retorne registos, demonstrando a não existência de registos em duplicado na tabela FACT_LEILAO_PONTOS
Acções de execução	Executar um carregamento inicial para o mês 200804 e guardar os resultados na tabela FACT_LEILAO_PONTOS_CI_04 Executar a seguinte query: *SELECT COUNT(*) FROM(* *SELECT a.COD_PONTOS_LEILAO, a.DATA_LICITACAO,* *a.COD_PONTOS_ESTADO_LICITACAO, a.NUMT, a.DATA_MODIFICACAO* *FROM FACT_LEILAO_PONTOS_CI_04 a* *GROUP BY a.COD_PONTOS_LEILAO, A.DATA_LICITACAO,* *a.COD_PONTOS_ESTADO_LICITACAO, a.NUMT, a.DATA_MODIFICACAO* *HAVING COUNT(*) >1)*

TESTE 7 – Reprocessamento	
Descrição	Pretende-se garantir que aquando de um reprocessamento, os dados carregados na tabela estejam correctos
Pré-requisitos	Correcta execução do teste 6
Dados de teste	FACT_LEILAO_PONTOS, FACT_LEILAO_PONTOS_CM_05 e FACT_LEILAO_PONTOS_RP_05
Resultados esperados	Espera-se que os resultados obtidos aquando de um reprocessamento estejam correctos, ou seja, que o número de registos existentes nas tabelas usadas para o teste seja o mesmo
Acções de execução	Executar o processo de carregamento inicial para o mês N. Executar o processo de carregamento mensal para o mês N+1. Guardar os dados na tabela: FACT_LEILAO_PONTOS_CM_05 Executar o processo de reprocessamento para o mês N. Executar o processo de reprocessamento para o mês N+1. Guardar os dados na tabela: FACT_LEILAO_PONTOS_RP_05 Comparar as tabelas, verificando se o número de registos é o mesmo

TESTE 8 – Contagem de duplicados após reprocessamento	
Descrição	Pretende-se garantir a não existência de registos em duplicado na tabela FACT_LEILAO_PONTOS após um reprocessamento
Pré-requisitos	Correcta execução do teste 7
Dados de teste	FACT_LEILAO_PONTOS_RP_05
Resultados esperados	Espera-se que a query de teste não retorne registos, demonstrando a não existência de registos em duplicado na tabela FACT_LEILAO_PONTOS após um reprocessamento
Acções de execução	Executar a seguinte query: *SELECT COUNT(*) FROM(* *SELECT a.COD_PONTOS_LEILAO, a.DATA_LICITACAO,* *a.COD_PONTOS_ESTADO_LICITACAO, a.NUMT, a.DATA_MODIFICACAO* *FROM FACT_LEILAO_PONTOS_RP_05 a* *GROUP BY a.COD_PONTOS_LEILAO, A.DATA_LICITACAO,* *a.COD_PONTOS_ESTADO_LICITACAO, a.NUMT, a.DATA_MODIFICACAO* *HAVING COUNT(*) >1)*

- FIM DO DOCUMENTO -

Efectuadas as primeiras validações e testes ao projecto, foram detectados os seguintes aspectos os quais resultaram num primeiro *feedback* para a equipa de desenvolvimento (envio de um e-mail contendo todos estes pontos e anexado com o plano de testes):

- **Documentação (manual de operação)**
  - Deve ser incluído no capítulo 4.1 a referência ao *shell script* dsLeilaoPontosFactPrc.sh <AAAAMM>;
  - Devem ser criados os seguintes sinónimos no capítulo 5.9 (sinónimos e privilégios):

    ***user*@bd = DW_LEILOES@WCMAPROD**

    CREATE SYNONYM TB_LEILAO FOR VGNAB.TB_LEILAO;
    CREATE SYNONYM TB_LICITACAO FOR VGNAB.TB_LICITACAO;
    CREATE SYNONYM TBLIC_ESTADO FOR VGNAB.TBLIC_ESTADO;
- **Documentação (*checklist*)**
  - Capítulo 6.2 (criação de directorias): deve ser indicado em que máquina e user devem ser criadas as directorias;

- Capítulo 6.3 (distribuição das fontes): deve ser indicado se os itens são novos ou foram apenas alterados;
- Capítulo 6.6 (*tablespaces*): o nome dos tablespaces de dados e índices da tabela FACT_LEILAO_PONTOS não estão de acordo com as normas do manual de desenvolvimento. Devem ser corrigidos os nomes e alterados nos ficheiros .ddl e .idx da tabela;

- **Item LeiloesPontos.dsx (ficheiro datastage):**
  - No *job* datastage DS_FactPontosLicPrc a *flag "Pre 4.2 User Defined SQL Behavior"* deve estar desligada no OCI ORA_HIST_CARD de acordo com as normas do manual de desenvolvimento;
- **FACT_LEILAO_PONTOS.ddl (ficheiro *Data Definition Language*)**
  - Não existe nenhuma tabela de Referência com a descrição do campo COD_PONTOS_LICITACAO. Sugere-se a alteração do nome deste campo para ID_PONTOS_LICITACAO;
- **FACT_LEILAO_PONTOS.idx (ficheiro de criação de índices)**
  - De acordo com o manual de desenvolvimento, deve ser incluída a cláusula LOCAL na criação dos índices, visto que a tabela FACT_LEILAO_PONTOS é particionada (ao mês).

O gestor de projecto da equipa de desenvolvimento procedeu então ao *demote* do desenvolvimento para que as alterações fossem efectuadas, regressando o projecto ao estado de Integração. Feitas as correcções e promovido novamente o projecto para o estado de Testes, a equipa de testes procedeu às respectivas validações dos pontos referidos, bem como à execução dos casos de teste anunciados no plano de testes. Dessas execuções, resultou a elaboração do seguinte relatório de testes:

RELATÓRIO DE TESTES DE VERFICAÇÃO

1. Sumário dos Testes:

Data de Início	**2008-05-28**	
Data de Fim	**2008-05-31**	
Número de Testes: 8	**Executados**	**8**
	Passados	**8**
	Falhados	**0**

2. Casos de Teste:

TESTE 1 – Distribuição dos itens e preparação do ambiente de trabalho		
Execução 1	**Data**	2008-05-28
	Resultados	Foi possível criar e distribuir todos os pontos definidos pela checklist de entrada em produção
Aprovação/Decisão	Teste executado com sucesso	

TESTE 2 – Execução do processo e verificação dos logs		
Execução 1	**Data**	2008-05-29
	Resultados	**dsLeilaoPontosRefExt.sh.200805.log** Processo dsLeilaoPontosRefExt.sh 200805 iniciado. Thu May 29 14:16:47 WEST 2008 Passo 1: Definicao de variaveis. Thu May 29 14:16:47 WEST 2008 Passo 2: Preparacao da area de trabalho. Thu May 29 14:16:47 WEST 2008 Passo 3: Verificacoes de ficheiros. Thu May 29 14:16:47 WEST 2008 Passo 4: Obter o mes anterior. Thu May 29 14:16:47 WEST 2008 Passo 5: Obter o ultimo dia do mes anterior. Thu May 29 14:16:47 WEST 2008 Passo 6: Compilacao dos jobs DataStage. Thu May 29 14:16:47 WEST 2008 Passo 7: Chamar o mapeamento de Processamento. Thu May 29 14:16:52 WEST 2008 Passo 8: Limpeza da Area de Trabalho. Thu May 29 14:17:01 WEST 2008 Processo dsLeilaoPontosRefExt.sh terminado com sucesso. Thu May 29 14:17:01 WEST 2008 (a título de exemplo, apenas foi apresentado um dos logs do processo)
Aprovação/Decisão	Teste executado com sucesso	

TESTE 3 – Verificação do correcto preenchimento da tabela FACT_LEILAO_PONTOS		
Execução 1	Data	2008-05-30
	Resultados	**Total de leiloes nas fontes:** *SELECT COUNT(*) FROM (* *SELECT * FROM (* *SELECT a.ID FROM TB_LICITACAO A* *WHERE a.LEILAO_ID_FK IN (* *SELECT b.ID FROM TB_LEILAO b WHERE ESTADO=1* *))* *UNION ALL* *SELECT * FROM (* *SELECT a.ID FROM TB_LICITACAO A* *WHERE a.LEILAO_ID_FK IN (* *SELECT b.ID FROM TB_LEILAO b WHERE ESTADO <> 1* *)))* *-- Result: 83 rows* **Leiloes "activos" nas Fontes:** *SELECT COUNT(*) FROM (* *SELECT * FROM (* *SELECT a.ID FROM TB_LICITACAO a* *WHERE a.LEILAO_ID_FK IN (* *SELECT b.ID FROM TB_LEILAO b WHERE ESTADO = 1)))* *-- Result: 40 rows* **Tabela FACT_LEILAO_PONTOS:** *SELECT COUNT(*) FROM FACT_LEILAO_PONTOS* *-- Result: 40 rows* Tal como era pretendido, foram carregados para a tabela FACT_LEILAO_PONTOS os registos com estado = 1
Aprovação/Decisão	Teste executado com sucesso	

TESTE 4 – Verificação do correcto preenchimento das tabelas de Referência		
Execução 1	Data	2008-05-30
	Resultados	**Tabela REF_LEILAO_PONTOS_ESTADO:** Carregamento efectuado correctamente da fonte TBLIC_ESTADO: *SELECT COUNT(*) FROM (* *SELECT a.ID, a.NOME, a.DESCRICAO FROM TBLIC_ESTADO a)* -- Result: 8 rows *SELECT COUNT(*) FROM REF_LEILAO_PONTOS_ESTADO* -- Result: 8 rows **Tabela REF_LEILAO_PONTOS:** Carregamento efectuado correctamente da fonte TB_LEILAO: *SELECT ID FROM TB_LEILAO WHERE ESTADO = 1 AND (DATA_MODIF_ESTADO < TO_DATE ('20080601', 'YYYYMMDD'))* *--Result: 3, 6, 5, 7, 10, 12* *SELECT COUNT(*) FROM REF_LEILAO_PONTOS WHERE COD_PONTOS_LEILAO IN (3, 6, 5, 7, 10, 12)* *--Result: 6 rows* *SELECT COUNT(*) FROM REF_LEILAO_PONTOS* *--Result: 6 rows* **Tabela REF_LEILAO_PONTOS_TIPO:** Carregamento efectuado correctamente do ficheiro REF_PONTOS_TIPO_LEILAO.txt: */directoria/filesystem> wc -l REF_PONTOS_TIPO_LEILAO.txt* *-- Result: 2* *SELECT COUNT(*) FROM REF_LEILAO_PONTOS_TIPO* *-- Result: 2 rows* Após análise aos registos resultantes das queries foi possível verificar o correcto preenchimento das tabelas de Referência
Aprovação/Decisão	Teste executado com sucesso	

TESTE 5 – Integridade Referencial		
Execução 1	Data	2008-05-30
	Resultados	*SELECT COD_PONTOS_LEILAO FROM FACT_LEILAO_PONTOS* *MINUS* *SELECT COD_PONTOS_LEILAO FROM REF_LEILAO_PONTOS* *--Result: 0 rows* *SELECT COD_PONTOS_ESTADO_LICITACAO FROM FACT_LEILAO_PONTOS* *MINUS* *SELECT COD_PONTOS_ESTADO_LICITACAO FROM REF_LEILAO_PONTOS_ESTADO* *--Result: 0 rows* *SELECT COD_CARD FROM FACT_LEILAO_PONTOS* *MINUS* *SELECT COD_CARD FROM REF_CARD* *--Result: 0 rows* *SELECT COD_U_CARD FROM FACT_LEILAO_PONTOS* *MINUS* *SELECT COD_U_CARD FROM REF_UNIC_CARD* *--Result: 0 rows* As queries usadas para teste retornaram zero registos como resultado, demonstrando que todos os códigos existentes na tabela Factual estão presentes nas respectivas tabelas de Referência
Aprovação/Decisão	Teste executado com sucesso	

TESTE 6 – Contagem de duplicados após carregamento inicial		
Execução 1	Data	2008-05-30
	Resultados	*SELECT COUNT(*) FROM(* *SELECT a.COD_PONTOS_LEILAO, a.DATA_LICITACAO,* *a.COD_PONTOS_ESTADO_LICITACAO, a.NUMT,* *a.DATA_MODIFICACAO* *FROM FACT_LEILAO_PONTOS_CI_04 a* *GROUP BY a.COD_PONTOS_LEILAO, A.DATA_LICITACAO,* *a.COD_PONTOS_ESTADO_LICITACAO, a.NUMT,* *a.DATA_MODIFICACAO*

		HAVING COUNT(*) >1) *-- Result: 0 rows* A query usada para teste não retornou qualquer registo, demonstrando assim a não existência de registos em duplicado na tabela FACT_LEILAO_PONTOS após um carregamento inicial
Aprovação/Decisão	Teste executado com sucesso	

TESTE 7 – Reprocessamento		
Execução 1	**Data**	2008-05-30
	Resultados	Executou-se o processo de carregamento inicial para o mês 200804. Executou-se o processo de carregamento mensal para o mês 200805. Guardou-se um backup da tabela: *CREATE TABLE FACT_LEILAO_PONTOS_CM_05 AS* *SELECT * FROM FACT_LEILAO_PONTOS* Executou-se o processo de reprocessamento para o mês 200804. Executou-se o processo de reprocessamento para o mês 200805. Guardou-se um backup da tabela: *CREATE TABLE FACT_LEILAO_PONTOS_RP_05 AS* *SELECT * FROM FACT_LEILAO_PONTOS* Executou-se as seguintes queries: *SELECT * FROM FACT_LEILAO_PONTOS_RP_05* *MINUS* *SELECT * FROM FACT_LEILAO_PONTOS_CM_05* *--Result: 0 rows* *SELECT * FROM FACT_LEILAO_PONTOS_CM_05* *MINUS* *SELECT * FROM FACT_LEILAO_PONTOS_RP_05* *--Result: 0 rows* *SELECT COUNT(*) FROM FACT_LEILAO_PONTOS_CM_05* *-- Result: 40 rows* *SELECT COUNT(*) FROM FACT_LEILAO_PONTOS_RP_05* *-- Result: 40 rows* Os resultados obtidos demonstram que após um reprocessamento os dados obtidos nas tabelas estão correctos
Aprovação/Decisão	Teste executado com sucesso	

TESTE 8 – Contagem de duplicados após Reprocessamento		
Execução 1	Data	2008-05-30
	Resultados	*SELECT COUNT(*) FROM (* *SELECT a.COD_PONTOS_LEILAO, a.DATA_LICITACAO,* *a.COD_PONTOS_ESTADO_LICITACAO, a.NUMT,* *a.DATA_MODIFICACAO* *FROM FACT_LEILAO_PONTOS_RP_05 a* *GROUP BY a.COD_PONTOS_LEILAO, A.DATA_LICITACAO,* *a.COD_PONTOS_ESTADO_LICITACAO, a.NUMT,* *a.DATA_MODIFICACAO* *HAVING COUNT(*) >1)* *-- Result: 0 rows* É possível verificar que após o processo de reprocessamento os dados obtidos na tabela Factual estão correctos
Aprovação/Decisão	Teste executado com sucesso	

3. Decisão:

Decisão	O projecto está pronto para entrada em Produção

- FIM DO DOCUMENTO -

Finalizado o relatório de testes, procedeu-se ao envio de um último *feedback* para a equipa de desenvolvimento, desta vez com a informação de que o projecto reunia todas as condições para ser promovido. Assim sendo, procedeu-se ao *promote* do mesmo para o nível imediatamente acima (Espera), ficando o projecto a aguardar a sua entrada em produção. Desta feita, a equipa de sistemas de produção procedeu posteriormente à instalação do projecto no ambiente de produção sem que ocorresse qualquer tipo de problema.

7.3 FERRAMENTAS UTILIZADAS

Algumas das ferramentas utilizadas para a aplicação deste caso prático foram:

- ***TOAD for Oracle*** – Aplicação de *software* para desenvolvimentos em SQL e administração de bases de dados. Utilizado principalmente para a preparação do ambiente de testes e execução dos casos de teste.
- ***FileZilla*** – Ferramenta de transferência de ficheiros por FTP (*File Transfer Protocol*). Utilizada para preparação do ambiente de testes.
- ***UltraEdit*** – Ferramenta para criação e edição de ficheiros. Foi usada principalmente para comparações entre diferentes versões de itens.
- ***Harvest*** – Ferramenta de controlo de versões. Permite guardar o histórico das versões dos itens, bem como fazer o controlo do ciclo de desenvolvimento dos projectos. Neste caso foi usada para associar os itens ao projecto e para proceder a *demotes* e *promotes* do projecto entre os vários estados.
- ***Ascential DataStage*** – Ferramenta de ETL. É utilizada para desenvolver processos de extracção, transformação e carregamento de dados. É uma das mais conhecidas e utilizadas ferramentas de ETL do mundo empresarial. Neste caso prático, foi utilizada para importar novos *jobs* e controlar o carregamento de dados para as tabelas.
- **ZOC** – Esta ferramenta é um cliente e emulador de terminal Telnet/SSH. Foi utilizada para aceder às máquinas destinadas para testes, permitindo as execuções dos processos.
- **PL/SQL** – É uma extensão da linguagem de programação SQL padrão. Foi utilizada para desenvolvimento de alguns dos itens do projecto bem como para o desenvolvimento de muitas das acções de execução dos casos de teste.
- ***Shell Script*** – Esta linguagem de *Scripting* foi usada para desenvolvimento de alguns dos itens do projecto.
- **AWK** – Linguagem de programação utilizada para processamento de dados baseados em texto. Foi usada conjuntamente com a linguagem *Shell Script* em alguns dos itens do projecto.

8 Conclusões

8.1 Apreciação Crítica / Avaliação

A experiência de integração numa equipa de testes de uma empresa de telecomunicações permite que seja efectuada esta apreciação crítica a todo o processo de testes a sistemas de *Data Warehouse*:

- Visto tratar-se de uma grande empresa, existem várias equipas a trabalhar provenientes de outras empresas. Por vezes, o relacionamento entre equipas pode não funcionar na perfeição, originando perdas de tempo desnecessárias e prejudiciais para o desenvolvimento dos projectos. A boa interacção entre equipas é fundamental, e deve ser privilegiada para que se obtenha uma resposta mais rápida às questões levantadas.
- Teoricamente, na fase de planeamento/enquadramento, devem ser feitas reuniões entre equipas de desenvolvimento e equipas de testes para esclarecimento de dúvidas nas especificações técnicas do projecto. Muitas vezes não são feitas essas reuniões, originando falta de visibilidade e envolvimento nos processos de engenharia de requisitos.
- No mercado altamente competitivo em que vivemos, as decisões fazem-se consoante a informação disponível (o mais actualizada possível) e são tanto mais eficazes quanto melhor for essa informação. A necessidade de urgência de entrada em produção dos projectos implica que por vezes não sejam efectuados tantos testes quanto seriam necessários. Além disto, o engenheiro de testes muitas vezes não tem tempo para se integrar totalmente nos projectos. São diversos projectos a testar (por vezes em simultâneo), complexos e que exigem mais tempo para integração do que o disponível.
- Algumas vezes, existe uma certa dificuldade em conseguir aplicar os requisitos de teste aos requisitos do negócio, resultante também um pouco da falta de visibilidade e envolvimento no projecto.
- Aquando de um *promote* após correcções efectuadas a um projecto, muitas vezes a equipa de testes não é correctamente notificada das reais alterações que foram efectuadas. Esta situação poderá implicar ao erro, ou até mesmo a um novo *demote* ao projecto. Além disso, muitas das

alterações são efectuadas sem ter em consideração o impacto causado nos testes.

- Na fase de preparação do ambiente de testes é necessário solicitar às equipas de sistemas de produção a replicação dos objectos necessários aos testes. Estas solicitações implicam mais perda de tempo. Algumas acções (tais como, o *export* de ficheiros ou o *export* de tabelas) poderiam ser feitas por vezes pela própria equipa de testes, sem comprometer o correcto funcionamento das bases de dados e *filesystem* do ambiente de produção (por exemplo através do fornecimento apenas de permissões de leitura a determinados users da base de dados).
- O perfil comportamental de um engenheiro de testes deve ser tão importante como o conhecimento técnico necessário. No mercado actual não existem muitas pessoas com conhecimentos em testes e o motivo principal é que os cursos de graduação não possuem disciplinas específicas de teste de *software*. Um novo elemento que integre uma equipa de testes, pode no início implicar alguns problemas à equipa, como por exemplo o atraso nas entradas em produção de projectos, o acumular de projectos no estado de Testes ou até a desmotivação dos restantes elementos da equipa de testes. Apesar disso, a longo prazo este novo reforço irá compensar em muito. É um facto que a experiência em testes pode ser adquirida, sendo necessário apenas que seja fornecida essa oportunidade.

8.2 PRINCIPAIS DESENVOLVIMENTOS

Os principais desenvolvimentos testados e validados no âmbito deste estágio curricular basearam-se no seguinte:

- Carregamento de ficheiros diários com informação de níveis de cobrança provenientes de uma determinada máquina remota para tabelas de *Staging Area*.

- Criação de um novo modelo no *Data Warehouse* com informação mensal de cartões associados a uma determinada campanha (a informação era proveniente dos sistemas operacionais).
- Criação de uma nova tabela de *Staging Area* para disponibilizar no *DataWarehouse* um serviço para wi-fi e banda larga.
- Disponibilização no *Data Warehouse* de novos tarifários associados a um determinado serviço, sendo necessária a criação de uma nova tabela de *Staging Area*.
- Alteração de estruturas de tabelas de Factos e de tabelas Agregadas para passarem a incluir novos campos necessários à disponibilização de novas informações importantes ao negócio.
- Integração de novos tipos de ficheiros no *Data Warehouse* garantindo a sua correcta distribuição por diversos modelos.
- Disponibilização de informação relativa a utilizadores de um determinado serviço, bem como dos dados desse serviço.
- Alteração do processo de extracção de um determinado desenvolvimento de forma a optimizar essa extracção em termos de tempo.
- Carregamento de fontes relevantes dos sistemas operacionais (com base em tabelas Informix) para a *Staging Area*, de forma a disponibilizar informação acerca de um determinado serviço. Foi necessária a criação de três novas tabelas de *Staging Area*.
- Disponibilização de novas tabelas de Factos (diárias, semanais e mensais) com informação de adesões a campanhas e desactivações de campanhas. Disponibilização de novas tabelas Agregadas (também diárias, semanais e mensais) de campanhas. Disponibilização de novas tabelas de Referência.
- Enriquecimento de tabelas de Referência com a inclusão de novas dimensões com base num ficheiro de *input* actualizado.

8.3 Conclusões Finais

Na época em que vivemos, o cliente acede rapidamente à informação que pretende tendo diversas possibilidades de escolha. Uma empresa tem que ter conhecimento do comportamento e das necessidades dos seus clientes de modo a puder antecipar situações de insatisfação. Isto para manter o cliente e evitar o *Churn*.

Num futuro próximo, todas as grandes organizações terão o seu *Data Warehouse* e um conjunto de aplicações e sistemas que acedem a esse repositório, lucrando da grande riqueza que existe na informação. Já estamos até na época do "*Webhousing*" (repositório com informação sobre o comportamento do cliente na Internet).

Um Data Warehouse é caro e, como tal, quando se pensa em desenvolver um projecto de *Data Warehouse* há que definir bem a informação que o sistema vai disponibilizar. Muitos projectos de *Data Warehouse* falham. Alguns porque nunca chegam a entrar em produção ou então apesar de entrarem, a informação que produzem não era a essencial, não resolvendo as necessidades da empresa.

Um projecto de *Data Warehouse* nunca pode ser dado como concluído dada a necessidade de se adaptar continuamente às necessidades da empresa. Surgem sempre novos requisitos de negócio e novas funcionalidades a serem implementadas e só através desta evolução contínua se pode trazer os benefícios previstos à performance empresarial.

A aplicação de metodologias de qualidade e testes assumem-se como essenciais para a manutenção e optimização do processo de *Data Warehouse*. Investir em testes é investir na qualidade da informação. Nas últimas décadas a qualidade de *software* tem vindo a ser uma área cada vez mais importante e dada a necessidade de melhorar esta qualidade, tem vindo a crescer a área de Testes de *Software*. O interesse pela actividade de testes aumenta à medida que a falta destes testes influenciam directamente o custo final de um *software* produzido. Dado isto, as exigências por *software* com maior qualidade têm

motivado a definição de novos métodos e técnicas para que os desenvolvimentos atinjam os elevados padrões de qualidade exigidos.

9 Bibliografia

(s.d.). Obtido em 16 de 9 de 2008, de Wikipedia: http://pt.wikipedia.org/wiki/Teste_de_software

BREITNER, C. A. (1997). *"Data Warehousing and OLAP: Delivering Just-In-Time Information for Decision Support". In: Proceedings of the 6th International Workshop for Oeconometrics, 6(4).* Karlsruhe, Germany.

Caldeira, C. P. (2008). In *Data Warehousing Conceitos e modelos com exemplos práticos.* Lisboa: Sílabo.

COME, G. (2001). *Contribuição ao estudo da implementação de data warehousing: Um caso no sector de telecomunicações. Dissertação (Mestrado em Administração) - Curso de Pós-graduação em Administração, Universidade de São Paulo.* São Paulo.

FALTYS, J. (2000). *Rules-Based Software for Telecommunications Targeted Marketing.* DM review.

Gardner, S. R. (1998). *Building the Data Warehouse.* New York, USA: ACM.

IDW. (2003). *IDW Integration, Development and Warehousing*. Obtido em 19 de 091 de 2008, de http://www.idw.pt

INMON, W. H. (1996). *Building the Data Warehouse.* New York: John Wiley & Sons.

Kimbal, R. e. (1998). *The Data Warehouse Lifecycle Toolkit.* New York: John Wiley & Sons .

MALLACH, E. G. (2000). *Decision support and data warehouse systems.* Boston: McGraw-Hil.

Matisson, R. (1997). *Data Warehousing and Data Mining for Telecomunications.* Boston. London.

MOODY, D. L. (2000). *"From Enterprise Models to Dimensional Models: A Methodology for Data Warehouse and Data Mart Design".* Stockholm, Sweden.

PETERS, R. (2000). *Data Warehousing in the Telecomunications Services Industry.* DM Review.

POWER, D. J. (2000). *Decision Support Systems Hyperbook. Cedar Falls, IA: DSSResources.COM.* Obtido de PDF version, http://dssresources.com/dssbook/.

RUFINO, R. (1998). *Data Warehousing - CTBC Telecom.*

Shams, K. F. (2001). *Data Warehousing: "Toward Knowledge Management", Topics in Health Information Management.*

TAURION, C. (1997). *Data Warehouse: Estado de Arte e Estado de Prática. Developers Magazine.*

THOMSON, E. (2002). *Construindo Sistemas de Informações Multidimensionais. 2ª ed.* São Paulo: Campus.

WATTERSON, K. (1998). *"Second Generation Data". SunExpert Magazine (Oct), pp. 58-65.*

William H. Inmon, C. I. (1996). *Building the Operational Data Store.* John Wiley & Sons.