INSTITUTO SUPERIOR DE CIÊNCIAS DO TRABALHO E DA EMPRESA
DEPARTAMENTO DE CIÊNCIAS E TECNOLOGIAS DE INFORMAÇÃO

O PROCESSO DE REFRESCAMENTO NOS SISTEMAS DE *DATA WAREHOUSE*

GUIÃO DE MODELAÇÃO CONCEPTUAL DA TAREFA DE EXTRACÇÃO DE DADOS

CARLOS MANUEL ROGADO QUINTINO DA COSTA PAIVA

DISSERTAÇÃO SUBMETIDA COMO REQUISITO PARCIAL PARA OBTENÇÃO DO GRAU DE

MESTRE EM GESTÃO DE SISTEMAS DE INFORMAÇÃO

ORIENTADOR:

PROFESSOR DOUTOR PEDRO RAMOS

JULHO, 2006

RESUMO

Nos últimos anos, os Sistemas de *Data Warehouse* (SDW) têm sido os sistemas de apoio à decisão mais utilizados nas organizações, integrando dados de diferentes fontes nos Repositórios de *Data Warehouse* (RDW).

Com o decorrer do tempo de funcionamento do sistema, coloca-se o problema do refrescamento, entendido como o problema de assegurar que os conteúdos dos RDW são periodicamente refrescados, de modo a reflectirem as alterações que ocorrem nos dados das fontes que lhes servem de base.

Esta dissertação propõe uma abordagem que tem como objectivos principais tornar explícito e documentar o problema do refrescamento e apresentar um guião de modelação conceptual da tarefa de extracção de dados que possa enriquecer as fases subsequentes de desenho para a especificação formal do processo de refrescamento.

São dois os contributos desta dissertação. Primeiro, providencia um quadro detalhado sobre o problema do refrescamento que inclui os conceitos e questões fundamentais que permitem caracterizar os SDW, na perspectiva das funcionalidades no apoio à decisão, das abordagens de integração de fontes de dados e dos componentes da arquitectura, os constrangimentos e tarefas que compreendem o processo de refrescamento, as principais abordagens disponíveis na literatura. Segundo, propõe um guião de apoio à modelação conceptual da tarefa de extracção de dados, com base na UML, apresentando os passos que devem ser seguidos pelo designer e disponibilizando as construções que permitem representar os dados que se extraem das fontes, de acordo com as regras que permitem isolar e extrair os dados relevantes para a tomada de decisão.

PALAVRAS-CHAVE: *Data Warehouse*, Refrescamento, Modelação Conceptual, Extracção de dados

ABSTRACT

Data Warehouse Systems (DWS) have become very popular in the last years for decision making, by integrating data from internal and external sources into data warehouse stores.

As times advances and the sources from which warehouse data is integrated change, the data warehouse contents must be regularly refreshed, such that warehouse data reflect the state of the underlying data sources.

This dissertation proposes an approach which main goals are to explicit and document the data warehouse refreshment problem and to present a guidelines for the conceptual modelling of data extraction in order to enrich the subsequent design steps for the formal specification of the refreshment process.

The contributions of our approach are twofold. First, it provides a detailed outline of data warehouse refreshment problem, including the main concepts and issues that characterise the general domain of the DWS, such as decision making functionalities, data sources integration approaches and architecture and, the refreshment tasks and constraints as well as the main approaches. Second, it proposes a guidelines for an UML conceptual modelling of data extraction, by giving the sequence of steps for a designer to follow, the modelling constructs for the definition of extracting data, according to the rules that must be accomplished for extracting relevant data.

Keywords: Data Warehouse, Refreshment, Conceptual Modelling, Data Extract

AGRADECIMENTOS

Esta dissertação foi preparada num período de consolidação na carreira profissional, exigindo uma permanente articulação entre as actividades de pesquisa e as actividades profissionais, abdicando, muitas vezes, da vida pessoal e social. Quero, por isso, expressar os meus agradecimentos a todos quantos contribuíram para tornar possível este trabalho.

Aos amigos, pelo apoio e por continuarem a adiar contar comigo para os momentos de lazer.

À minha irmã, por ter tornado sempre felizes e bonitos os curtos momentos de família, trazendo a sua presença carinhosa e materna e a alegria e vida do Guilherme e do Bernardo.

Ao meu pai, pelo apoio e motivação, por compreender as minhas longas ausências, num momento em que tanto precisava da minha companhia.

À minha mãe, por me ensinar a pensar, a pesquisar e a construir, pelo apoio e longas horas que me dedicou.

À Sissi, pelo apoio, companheirismo e compreensão.

Devo ainda a cortesia de investigadores na área dos SDW que tão prontamente responderam aos meus e-mail, criticando, dando sugestões e até mesmo fornecendo textos não publicados. Destacam-se Ronald Stamper, um dos autores do FRISCO, Panos Vassiliadis, da Universidade Técnica de Atenas e colaborador do Projecto DWQ, Sergio Luján-Mora, da Universidade de Alicante, Maurizio Lenzerini, da Universidade de Roma e também colaborador do Projecto DWQ, Lina Nemuraitè e Florian Heidenreich, da Universidade de Dresden.

À Professora Helena Galhardas por me ter recebido e ter-me ajudado a esclarecer o domínio da Extracção, Transformação e Carregamento.

Aos meus professores por tudo quanto me ensinaram durante a parte curricular do Mestrado. Um agradecimento especial à Professora Doutora Maria José Trigueiros, pelos ensinamentos na área dos sistemas de apoio à decisão.

Finalmente, ao meu orientador, Professor Doutor Pedro Ramos, pelas tantas e intermináveis horas de trabalho conjunto, mas, sobretudo, pelos tempos de aprendizagem de conhecimentos e da prática de investigação reflexiva que me proporcionou. Obrigado por ter acreditado em mim.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

1 INTRODUÇÃO

Nos sistemas de *Data Warehouse*, o processo de refrescamento tem por objectivo a captura periódica das alterações ocorridas nos dados das fontes, sua preparação e propagação ao longo dos repositórios de *Data Warehouse*.[1] Neste capítulo faz-se a apresentação do problema, identificam-se as questões de pesquisa e definem-se os objectivos centrais desta dissertação (secção 1.1). Introduz-se uma visão geral sobre o problema e descrevem-se os principais contributos da abordagem adoptada nesta dissertação para a solução do problema do refrescamento (secção 1.2). Finalmente, descreve-se a metodologia de investigação (secção 1.3) e detalha-se a forma como o trabalho está estruturado, a partir da Introdução (secção 1.4).

1.1 Problema e Objectivos de Pesquisa

Perante um mercado fortemente competitivo e dinâmico, as organizações têm vindo a confrontar-se com a necessidade de um sistema de informação capaz de transformar um grande volume de dados dispersos por uma diversidade de fontes em informação relevante que possa melhorar os processos de decisão e permitir uma eficiente análise, planeamento e controlo das actividades. Desde a clássica obra de Inmon (1996), publicada em 1992, que os Sistemas de *Data Warehouse* (SDW) têm sido um dos sistemas de apoio à decisão mais utilizados pelos gestores no acompanhamento do estado e evolução das actividades das organizações, no conhecimento de padrões de tendências, na projecção de previsões (Arnott e Pervan 2005: 72).

Um SDW compreende tecnologias, arquitecturas, algoritmos, modelos, ferramentas e aspectos organizacionais e de gestão que cooperam no sentido de integrar dados de diferentes fontes internas e externas à organização, por forma a publicar informação relevante para o apoio à tomada de decisão (Vavouras 2002: 1-2).

Na literatura, o conceito de *data warehouse* tem tomado diversas acepções, embora seja amplamente aceite a definição introduzida, em 1992, por Inmon. Para este

[1] Entre as várias designações colhidas na literatura em língua inglesa (*refreshment*, *freshment*, *propagation and refresh*, *Extracting, Transforming and Loading* (ETL), *maintenance*), optou-se por *refreshment*, aqui traduzido por refrescamento, por ser a designação mais divulgada recentemente para expressar a natureza do processo.

autor, *data warehouse* é o componente tecnológico central do sistema e é entendido como um repositório de dados para o apoio à decisão, cujas características básicas são: (s*ubject oriented*) orientado a assuntos ou áreas de interesse da organização, como por exemplo produtos, clientes, vendas e compras; (*integrated*) integra dados de fontes internas e externas à organização, através de um processo que extrai, combina e traduz os dados de uma diversidade de fontes num formato uniforme e global de dados definido para o *data warehouse*; (*non-volatile*) os dados armazenados no *data warehouse* são estáveis e de acesso apenas para leitura; (*time variant*) e possuem uma dimensão temporal longa (Inmon 1996: 33-7).

No conceito de Inmon está implícita uma das características fundamentais dos SDW e que se consubstancia na separação entre, por um lado, as fontes de dados que gerem dados detalhados, atómicos e correntes, acedidos através de aplicações de *Online Transaction Processing* (OLTP) e, por outro, o *data warehouse* que disponibiliza dados integrados, consolidados e históricos que suportam as aplicações de *Online Analytical Processing* (OLAP). As fontes de dados documentam as operações diárias de inserção, actualização e eliminação de dados, variando a qualidade dos dados e as estruturas e semânticas que os suportam. Enquanto que o *data warehouse* assenta numa estrutura uniforme e global de dados que são de acesso apenas para leitura (Chaudhuri e Dayal 1997: 65-6).

Todavia, a literatura e a prática também revelam que os dados *warehouse* podem estar disponíveis num único repositório de dados ou distribuídos por uma hierarquia de camadas de dados com diferentes níveis de detalhe. Pode incluir desde o *Operational Data Store* (ODS) que serve de repositório intermédio que integra os dados detalhados e recentes extraídos das fontes, o *Data Warehouse* (DW global, central, primário ou corporativo) que guarda o histórico dos dados e dados de baixa sumarização e os repositórios locais, como as bases de dados para o processamento OLAP ou os *data marts* que guardam dados de elevada sumarização (Jarke e Quix 2003: 2-4). Sempre que não se torne relevante distinguir os vários repositórios, opta-se pela designação genérica de Repositórios de *Data Warehouse* (RDW).

Os SDW são, pois, sistemas complexos que podem compreender várias camadas de dados e que envolvem quatro actividades fundamentais: (1) a aquisição de dados que acautela a captura, preparação e propagação dos dados das fontes para os RDW; (2) a gestão do armazenamento e manipulação dos dados no ambiente *warehouse*; (3) o acesso aos dados *warehouse* que permite a diferentes utilizadores consultar, visualizar,

analisar e manipular os dados publicados, através de aplicações de consulta directa e emissão de relatórios e de ferramentas de OLAP e de *data mining*; (4) a gestão do sistema, em que os metadados constituem a camada semântica que suporta o desenho, o funcionamento, a utilização, a manutenção e a administração do sistema.

No desenvolvimento de um SDW, a primeira fase de aquisição de dados consiste no carregamento inicial dos RDW. Com o decorrer do tempo de funcionamento do sistema coloca-se um dos problemas mais críticos de solucionar: o problema do refrescamento, entendido como o problema de assegurar que os conteúdos dos RDW reflectem as alterações que ocorrem nos conteúdos das fontes de dados que lhes servem de base, de modo a providenciarem informação com qualidade para o apoio à decisão.

As diferentes camadas de dados, o tempo e o grande volume de dados envolvidos, a execução de múltiplas transacções com complexidades variadas, as exigências de detalhe analítico e de qualidade da informação e o conflito entre as transacções que refrescam os dados e as transacções que consultam os mesmos dados são aspectos que tornam o refrescamento dos RDW um problema crítico (Bouzeghoub et al. 2003: 51-2).

Trata-se de um problema, cuja solução exige o desenho e a execução de um conjunto de tarefas responsáveis por capturar, preparar e propagar, em tempo útil, as alterações ocorridas nos dados das fontes para os correspondentes dados armazenados no ambiente *warehouse*. As propriedades de autonomia, distribuição e heterogeneidade das fontes de dados e a importância de se identificar que dados devem ser extraídos, de que fontes de dados e quando tornam a extracção de dados a tarefa que, muitas vezes, consome mais tempo de desenho e de execução em todo o processo de refrescamento (Fan 2005: 31-3).

O trabalho apresentado nesta dissertação enquadra-se na área da investigação dos SDW, mais concretamente foca no problema do refrescamento dos RDW e explora questões associadas às tarefas necessárias e a como e quando refrescar os conteúdos dos RDW, dando particular atenção à tarefa de extracção de dados.

Os objectivos centrais consistem em tornar explícito e documentar o problema do refrescamento dos RDW, tal como está reflectido na investigação e na prática, e propor um guião de modelação conceptual da tarefa de extracção de dados que possa enriquecer as fases subsequentes de desenho para a especificação formal do processo de refrescamento dos RDW.

1.2 Visão Geral sobre o Refrescamento dos RDW

Nos SDW, a captura, preparação e propagação das alterações ocorridas nos dados das fontes ao longo dos RDW é um processo complexo que depende dos requisitos dos utilizadores e está sujeito a constrangimentos associados às fontes de dados e ao ambiente *warehouse*. O processo compreende as tarefas de extracção de dados das fontes, de limpeza das suas anomalias, de reconciliação das diferenças das estruturas e das semânticas e consequente integração dos dados num formato uniforme e consolidado definido para os dados *warehouse*, de cálculo dos dados derivados, de preservação do histórico e de carregamento dos RDW. Acresce ainda a calendarização das várias tarefas, bem como a decisão entre o carregamento total dos dados das fontes, em cada período de refrescamento, ou o carregamento incremental que permite a aquisição apenas das alterações ocorridas nos dados das fontes, entre dois períodos sucessivos de refrescamento (Vavouras 2002: 25-45).

A curva do conhecimento sobre o problema do refrescamento tem crescido a um ritmo muito lento e tem preocupado, quer a comunidade industrial, quer a comunidade científica (Vassiliadis et al. 2005a: 522-8). Práticos e investigadores reclamam a urgência de mais investimento científico, particularmente no domínio dos modelos conceptuais e lógicos e das metodologias de suporte de desenho, sob pena de se comprometer o sucesso de um projecto de SDW (ver Agosta 2002, Shin 2003, Rizzi e Song 2004: 62-3). Isto porque os custos, o tempo e o esforço despendidos no refrescamento dos RDW têm um peso muito elevado no total do projecto de um SDW (ver Inmon 1997, Kimball e Caserta 2004: 23).

No nível da indústria, o esforço tem sido no sentido de colocar no mercado ferramentas especializadas que tomam a designação genérica de ferramentas de ETC[2], destinadas a executar tarefas como a extracção de dados relevantes, a sua limpeza, transformação e carregamento nos RDW (Vassiliadis et al. 2005a: 522-3). Na sua maioria, têm-se revelado ferramentas complexas, caras e que ainda não solucionam com eficácia o problema específico do refrescamento (Vavouras 2002: 51).

[2] ETC (Extracção, Transformação e Carregamento) corresponde à tradução do acrónimo ETL (*Extracting, Transforming and Loading*), amplamente utilizado pela comunidade científica e pela indústria para designar a actividade de aquisição de dados e que considera tanto o processo de carregamento inicial como o processo de refrescamento.

No nível da investigação, o problema do refrescamento dos RDW tem sido abordado de diferentes perspectivas e colhido contributos fundamentais da pesquisa no domínio alargado do processo de ETC.

No domínio específico do refrescamento, destacam-se as abordagens de manutenção das vistas materializadas, tradicionalmente associadas aos projectos WHIPS e Squirrel que reduzem o refrescamento à actualização das vistas materializadas e propõem soluções assentes num conjunto de componentes de arquitectura, técnicas e algoritmos[3]; a abordagem do projecto europeu DWQ que introduz o conceito de *workflow* para representar conceptualmente o processo de refrescamento decomposto em várias tarefas e constrangimentos associados (Bouzeghoub et al, 1999) e cuja solução proposta assenta num *framework* global de desenho dos SDW, abrindo caminho às abordagens orientadas a modelos[4]; a abordagem do projecto SIRIUS que propõe uma arquitectura flexível de *middleware* para a modelação e execução do processo de refrescamento[5].

No domínio do processo de ETC, de realçar o projecto Arktos que desenvolve um *framework* de modelação e execução de cenários de ETC, disponibilizando as ferramentas Arktos e Arktos II.[6] No nível conceptual de modelação do processo de ETC, são raras excepções Vassiliadis et al (2002a), Simitsis e Vassiliadis (2003), Simitsis (2005), Trujillo e Luján-Mora (2003) e Luján-Mora et al. (2004), sendo ainda de referir a documentação informal do processo de ETC providenciada por Kimball et al. (1998) e Kimbal e Caserta (2004).

Não obstante os valiosos contributos e soluções propostas pelas diversas abordagens, os autores são unânimes em reconhecer que o problema do refrescamento dos RDW está longe de estar esclarecido e solucionado, denunciando um conjunto de questões ainda em aberto na investigação, mas necessárias à modelação e execução das

---

[3] O projecto *WareHouse Information Prototype Stanford* foi desenvolvido por investigadores da Universidade Norte-Americana de Stanford e o projecto Squirrel por investigadores da Universidade Norte-Americana do Colorado. Ver http://www-dbstanford.edu/warehousing/warehouse.html.

[4] *Foundations of Data Warehouse Quality* é um projecto europeu de investigação (1996-9) que contou com a colaboração de várias universidades e centros de investigação, dando origem a um volume significativo de obra publicada e de dissertações de doutoramento (Jarke e Vassiliou 1997). Ver http://www.dbnet.ece.ntua.gr/~dwq/.

[5] O projecto *Supporting Incremental Refreshment of Information WarehoUseS* foi desenvolvido por investigadores da Universidade de Zurique. Ver http://www.ifi.unizh.ch/dbtg/projects/sirius/sirius.html.

[6] O projecto Arktos é desenvolvido por investigadores da Universidade Técnica de Atenas, uma das parceiras no projecto DWQ. Ver http://www.dblab.ntua.gr/.

tarefas de refrescamento, designadamente no que refere à tarefa de extracção de dados das fontes, muito ausente na pesquisa.[7]

Destacam-se as diferentes disponibilidades de acesso e capacidades das fontes apoiarem a detecção das alterações ocorridas nos seus conteúdos (Engström 2003: 72, Bouzeghoub et al. 2003: 53-62), a heterogeneidade das semânticas e das estruturas das fontes de dados e os requisitos dos utilizadores (Simitsis e Vassiliadis 2003: 2).

Por outro lado, não se conhece nenhuma pesquisa que aborde o processo de refrescamento no nível conceptual de modelação. No domínio do processo de ETC, tem-se reconhecido o contributo da modelação conceptual para o sucesso de um projecto de SDW, logo na fase inicial do desenho (Luján-Mora et al. 2004: 191-2).

De facto, as metodologias de suporte e os modelos conceptuais baseados em conceitos próximos das percepções dos utilizadores constituem instrumentos essenciais de documentação e análise que facilitam a comunicação entre os *stakeholders*.[8] Suportam as subsequentes fases do desenho e apoiam o designer na especificação e redefinição do processo de refrescamento, sem procurar providenciar a solução técnica para a sua execução (ver Vassiliadis et al. 2002a, Simitsis e Vassiliadis 2003, Trujillo e Luján-Mora 2003, Luján-Mora et al. 2004, Simitsis 2005, Vassiliadis 2005b).

Do mesmo modo que os recursos de uma linguagem de modelação orientada a objectos, como é a *Unified Modeling Language* (UML, Booch et al. 2005), pode apoiar a construção de modelos que permitem representar, no nível conceptual, as várias tarefas de refrescamento e constrangimentos associados, independentemente da tecnologia utilizada na execução do processo (ver Trujillo e Luján-Mora 2003, Luján-Mora et al. 2004).

São esses entendimentos científicos e questões em aberto divulgadas na pesquisa e na prática que constituem os eixos centrais da abordagem adoptada nesta dissertação. Uma abordagem que trata o refrescamento como um processo complexo que deve ser decomposto em tarefas mais simples para permitir lidar com os requisitos dos utilizadores e com os constrangimentos impostos pelos repositórios de dados envolvidos e que se situa na linha das abordagens orientadas a modelos conceptuais no tratamento da tarefa de extracção.

---

[7] Questões também referidas pela prática (ver, por exemplo, Adelman et al. 2003, Kimball e Caserta 2004).

[8] *Stakeholders* é a designação genérica que tomam os indivíduos com um interesse particular no sistema, englobando os utilizadores, os decisores, os responsáveis pelos sistemas de informação, os proprietários dos dados, os programadores, os designers, etc. (Robertson e Robertson 1999).

A singularidade da abordagem adoptada revela-se no afunilamento dos níveis de análise que permite, num nível mais geral, tratar as características fundamentais dos SDW; num nível intermédio, centrar nas tarefas de refrescamento e nas diversas abordagens e soluções propostas; e, num nível mais específico, focalizar na tarefa de extracção de dados, através da abordagem da modelação orientada a objectos.

Os principais contributos desta dissertação são os seguintes:

1 Providencia um quadro detalhado que permite caracterizar o problema do refrescamento dos RDW:

1.1 Introduzindo o conjunto de conceitos básicos e questões centrais que caracterizam os SDW, na perspectiva das funcionalidades no apoio à decisão, da integração de fontes de dados e dos componentes da arquitectura;

1.2 Avaliando os constrangimentos e os requisitos dos utilizadores e descrevendo as tarefas de refrescamento;

1.3 Discutindo as abordagens sobre o problema do refrescamento dos RDW, numa perspectiva comparada e crítica e indo para além das sínteses disponíveis e já algo desactualizadas.[9]

2 Propõe um guião de apoio à modelação conceptual da tarefa de extracção de dados:

2.1 Recorrendo aos conceitos de uma linguagem de modelação funcional e amplamente utilizada, como é a UML, independentemente da tecnologia utilizada na execução da tarefa da extracção de dados (Booch et al.2005);

2.2 Utilizando os conceitos propostos pelo *FRamework of Information System COncepts* (FRISCO), para entender e organizar os aspectos a modelar (Falkenberg et al. 1998);

2.3 Descrevendo um exemplo corrente que ilustra a aplicação dos conceitos utilizados;

2.4 Apresentando os passos, as recomendações e as convenções que facilitam a actividade de modelação do designer;

2.5 Providenciando as construções que permitem representar os dados que se extraem das fontes, de acordo com as regras que permitem isolar e extrair os dados relevantes para a tomada de decisão. O modelo conceptual proposto providencia metadados passivos, isto é, não descreve o domínio modelado num

[9] Vejam-se as sínteses de Gatziu et al. (1998), Theodoratos e Bouzeghoub (1999 e 2001), Vassiliadis (2000b), Engström (2002), Vavouras (2002), Trujillo e Luján-Mora (2003), Vassiliadis et al. (2005a).

alto nível de detalhe técnico por forma a suportar a execução da tarefa de extracção de dados, mas constitui-se como documentação para apoiar a especificação dos metadados que activamente conduzem a execução do processo de refrescamento (Staudt et al. 1999a: 42-3).

1.3 Metodologia de Investigação

A estratégia metodológica consistiu: (1) na análise da bibliografia mais pertinente na área dos SDW, para detectar os principais conceitos e questões que caracterizam o domínio geral onde se localiza o problema de refrescamento dos RDW, e na avaliação crítica e comparada da literatura específica sobre o refrescamento e o ETC, para esclarecer o domínio do problema; (2) no estudo aprofundado dos recursos da UML e do FRISCO, quer em termos do entendimento dos conceitos, quer em termos da sua utilidade prática na definição do guião de modelação conceptual.

A pesquisa bibliográfica foi realizada em bibliotecas da especialidade e, mais intensamente, na Internet, para além das obras de referência adquiridas para a biblioteca pessoal.

1.4 Estrutura do Trabalho

A parte deste trabalho que se segue à Introdução está organizada em cinco capítulos, seguindo a lógica do afunilamento das escalas de análise. O capítulo 2 introduz os conceitos básicos e questões centrais que permitem caracterizar os SDW. O capítulo 3 explora o domínio específico do processo de refrescamento dos RDW, descrevendo-se as tarefas e constrangimentos associados e no capítulo 4, discutem-se as principais abordagens sobre o refrescamento, dando-se particular atenção às abordagens orientadas a modelos. No capítulo 5, apresenta-se o guião de apoio à modelação conceptual da tarefa de extracção de dados. Finalmente, o capítulo 6 sumaria os principais resultados, reflecte sobre as limitações do estudo e aponta orientações para trabalhos futuros. Seguem-se a bibliografia e os apêndices. Os conteúdos das divisões de cada capítulo são referidos no início de cada um dos capítulos.

2 SISTEMAS DE *DATA WAREHOUSE*

Os SDW são sistemas complexos que integram diferentes fontes de dados e disponibilizam informação relevante para o apoio à decisão, garantindo meios de acesso simples, intuitivos e interactivos. O objectivo principal deste capítulo é introduzir um conjunto de conceitos básicos e questões centrais dos SDW que caracterizam o contexto geral do problema do refrescamento. Debatem-se as características e vantagens dos SDW como um sistema de apoio à decisão (secção 2.1) e apresentam-se as abordagens de integração de fontes de dados (secção 2.2). O quadro geral dos SDW é clarificado através da apresentação e análise dos conceitos e componentes básicos da arquitectura (secção 3.4), apresentando-se uma arquitectura típica (subsecção 2.2.1) e valorizando-se os RDW (subsecção 2.2.2), as abordagens de armazenamento para o processamento OLAP (subsecção 2.2.3), as ferramentas de OLAP e de *data mining* (subsecção 2.2.4) e o papel dos metadados na gestão e monitorização do sistema (secção 2.2.5).

2.1 Funcionalidades no Apoio à Decisão

A função última dos SDW é publicar informação relevante para suportar o processamento de consultas[10] directas e emissão de relatórios, análise complexa e descoberta de conhecimento, levadas a cabo por gestores, nos seus processos de decisão, incluindo-se, assim, na categoria dos Sistemas de Apoio à Decisão (SAD).

Os SAD constituem hoje um vasto e heterogéneo campo de pesquisa da disciplina dos Sistemas de Informação (SI) que estuda os sistemas que apoiam e informam os processos de decisão na gestão organizacional, contando com um grande volume de obra publicada e uma variada gama de sistemas, com um investimento significativo no domínio da indústria.[11]

[10] Consultas no sentido de interrogações (*queries*) que os utilizadores colocam, numa linguagem apropriada (linguagem de *query*), às bases de dados para obter informação útil sob a forma de resposta (Ramakrishnan e Gehrke 2003: 16).

[11] A importância dos SI nas organizações tem sido objecto de reflexão também no domínio da gestão e administração das organizações, sendo referências Ward et al. (1990), Keen (1991), Martin et al. (1994), Laudon e Laudon (2000). Numa crítica recente sobre este campo de pesquisa, Arnott e Pervan chamam a atenção para a necessidade de estudos de caso interpretativos e de um maior rigor científico, uma vez que as publicações revelam uma qualidade variável (2005: 67-8).

O conceito de SAD é muito amplo e polissémico, quer na literatura, quer no universo dos vendedores, dos consultores e até mesmo dos próprios gestores. Diz respeito a uma variedade de sistemas suportados por diferentes tecnologias e arquitecturas e com funcionalidades de complexidade variada (Erik 1997: 66-69). Abarca diversas designações e inclui desde sistemas personalizados até sistemas que suportam grupos de gestores num ambiente de rede cliente-servidor, uns focalizados nos dados, outros nos modelos, outros ainda nas comunicações (Power 2000: 4-5).

Por isso, os autores têm apresentado diferentes definições do conceito de SAD, consoante os aspectos considerados mais significativos, optando-se aqui pela definição proposta por Marakas, pelo facto de ser uma das definições que melhor sintetiza as principais características dos SAD, entendidos como sistemas com funcionalidades analíticas "sob o controlo de um ou mais agentes de decisão, que assiste na actividade de decisão, providenciando um conjunto organizado de ferramentas planeadas para estruturar porções da situação de decisão e melhorar a eficácia última do efeito da decisão" (1999: 31).[12]

Também as taxionomias apresentadas para darem conta da diversidade dos SAD são inúmeras e têm variado no tempo e em função das perspectivas e das abordagens dos autores.[13]

A taxionomia proposta por Alter é a mais divulgada na pesquisa sobre os SAD (1980) e, embora datada, revela-se ainda útil no entendimento de SAD actuais (Arnott e Pervan 2005: 70). A partir de sete categorias de componentes observadas, Alter classifica os SAD em duas classes, de acordo com os fundamentos técnicos: (1) a classe dos SAD orientados a modelos, que apenas estimam as consequências das acções, sem usar ou usando parcialmente o cálculo estatístico dos dados, e que oferecem cenários simulados de soluções óptimas ou providenciam sugestões para tarefas relativamente estruturadas, baseadas em regras ou modelos de capacidades analíticas variadas, e com uma interface de utilização simples; (2) a classe dos SAD orientados a dados, que calculam as consequências das acções devidamente planeadas, com base em cálculo estatístico, e que se apoiam em modelos intuitivos de dados disponibilizados num

[12] Tradução livre do autor.

[13] Para uma discussão sobre as várias definições e taxionomias, ver, por exemplo, Mallach (1994 e 2000), Roth (1995), Dhar e Stein (1997), Drazdzel e Flynn (2000), Marakas (1999), Power (2000 e 2001), Turban et al. (2001), Chen (2002), Arnott e Pervan (2005).

formato adequado ao acesso imediato e à sua manipulação analítica no apoio à gestão nas organizações. Os SDW são classificados como SAD orientados a dados.

O surgimento de novos SAD e a proliferação de termos, definições e taxionomias levam Power a propor um *framework* que possa reorganizar o campo de pesquisa sobre os SAD (2001). A classificação assenta numa dimensão principal, o componente tecnológico dominante, e em três dimensões secundárias, os utilizadores alvo, os objectivos que o sistema cumpre e a tecnologia principal utilizada, permitindo identificar cinco classes de SAD: *Data-driven*, *Model-driven*, *Knowledge-driven*, *Document-driven* e *Comunications-driven* e *Group* (Power 2001: 432-6).

Na classificação de Power, os SDW são classificados como SAD da classe *Data-driven*, cujo componente tecnológico dominante é o DW global que oferece aos gestores informação de fácil acesso para o processamento de consultas directas e emissão de relatórios, acrescido das funcionalidades de OLAP. Por seu lado, as ferramentas de *data mining* são classificadas na classe *Knowledge-driven*, por providenciarem conhecimento sobre um domínio específico (Power 2001: 432-3).[14]

Pode, assim, inferir-se que os SDW podem ser classificados como SAD híbridos, termo usado pelo autor para se referir a sistemas que possuem outros componentes com função de SAD, para além do componente dominante (Power 2001: 432). De facto, os SDW podem incluir ferramentas que permitem análises complexas para o apoio à decisão, como as ferramentas de OLAP que organiza os dados, de acordo com múltiplas dimensões, e combina-os em agregados, mas também ferramentas de *data mining* que descobrem modelos e padrões de ocorrência em grandes volumes de dados detalhados.

Uma outra perspectiva para analisar os SAD tem-se centrado na evolução histórica dos fundamentos teóricos e tecnológicos associados às várias classes de SAD, no quadro das necessidades de informação das organizações no tempo, procedendo-se à comparação entre as suas funcionalidades e distinguindo-se esses sistemas com funcionalidades analíticas dos tradicionais sistemas operacionais de OLTP, que muito cedo apoiaram as tarefas de gestão, nas organizações.[15]

[14] No mesmo sentido o entendimento de Dhar e Stein (1997: 5, 30-50) e de Mallach (2000: 143).

[15] Para uma discussão sobre a evolução dos SAD, ver, por exemplo, Mallach (1994 e 2000), Roth (1995), Inmon (1996), Dhar e Stein (1997), Drazdzel e Flynn (2000), Marakas (1999), Power (2000 e 2001), Turban et al. (2001), Arnott e Pervan (2005).

A adopção desta perspectiva, com foco nos SDW, permite esclarecer as principais vantagens desta classe de SAD.

No início de 1960, as organizações começaram a informatizar muitos dos processos operacionais das suas actividades, servindo-se de bases de dados concebidas como repositórios para coleccionar e armazenar grandes volumes de dados detalhados. Essas bases de dados colocavam online as operações correntes e apoiavam a gestão dos processos operacionais, através de aplicações de OLTP que permitiam apenas o processamento de consultas simples e predefinidas para a extracção directa e rápida de dados detalhados e emissão de relatórios (Chen 2002: 43-5 e 52, Ramakrishnan e Gehrke 2003: 6 e 847).

A partir dos anos de 1970, com o desenvolvimento da teoria e da tecnologia, começam a surgir os SAD para preencher as novas necessidades de informação das organizações que passam a enfrentar os desafios da competição à escala global, da produtividade, da diversificação de produtos e serviços e das incertezas e restrições de tempo na realização das actividades. Os gestores passam a empenhar-se na rapidez da tomada de decisões apoiadas na informação, recorrendo a dados sobre a evolução das actividades que possam garantir o sucesso das acções planeadas para atingir as metas estrategicamente definidas e o controlo da eficácia global da organização, atendendo a factores internos e externos (Mallach 2000: 13-15).

Neste novo cenário, cresce o volume de dados detalhados armazenados sob diversas estruturas e semânticas e dispersos por múltiplas fontes, tornando-se os tradicionais sistemas operacionais de OLTP incapazes de satisfazer as exigências de uma análise eficiente e eficaz dos dados da organização que possa permitir avaliar a situação e evolução das actividades e melhorar o processo de decisão estratégica dos gestores (March e Heyner 2003).

Os SDW surgem, no início dos anos de 1990, como a solução ideal para responder a estas novas exigências de informação das organizações (Baumöl et al. 2000: 37-9). Na origem dos SDW está o desenvolvimento da teoria das bases de dados, fundamentalmente as bases de dados de OLAP e a modelação multidimensional e metodologias de desenho que convergiram no sentido da definição de arquitecturas modernas.[16]

[16] Esses aspectos são desenvolvidos na secção 2.4.

A Tabela 2.1 sintetiza as principais diferenças entre as Bases de Dados (BD) de OLTP e as Bases de Dados (BD) de OLAP, recorrendo a propostas de vários autores.[17]

Características	BD de OLTP	BD de OLAP
Função/Processamento	Operações diárias/OLTP	Suporte à decisão/OLAP
Utilizadores	Designer e administrador do sistema, operadores	Gestores, analistas
Desenho BD	Modelo entidade-relação, Orientado a aplicações	Modelo *estrela*/*floco-de-neve*, Orientado a assuntos
Dados	Operacionais Correntes Detalhados Atómicos	Analíticos Históricos Detalhados e sumariados Integrados
Uso	Repetitivo, rotina	*Ad-hoc*
Acesso	Leitura e escrita Transacções e processamento de consultas simples (1-3 tabelas)	Leitura Processamento de consultas e análises complexas (mais tabelas)
Requisitos Sistema	Consistência dos dados	Qualidade dos dados

Tabela 2.1 Principais diferenças entre BD de OLTP e BD de OLAP

As vantagens dos SDW assentam numa estrutura de informação que aproveita as funcionalidades das fontes de dados e incorpora novas funcionalidades. Inclui as perspectivas organizacionais e de gestão, extraí, combina e traduz os dados das fontes num formato uniforme e consolidado e carrega-os nos RDW, gere dados detalhados e dados sumariados, dados actualizados continuamente em tempo real e dados históricos filtrados periodicamente. Disponibiliza uma visão global e evolutiva das actividades da organização, providencia informação relevante e refrescada periodicamente para o apoio à decisão dos gestores e facilita a análise complexa e a descoberta de conhecimento, através de ferramentas especializadas (Bischoff 1997: 4-6).

2.2 Integração de Fontes de Dados

Se a função última dos SDW é providenciar informação relevante para o apoio à decisão dos gestores, uma actividade fundamental no funcionamento dos SDW é a aquisição dos dados do ambiente operacional orientado a aplicações para o ambiente *warehouse* orientado a assuntos. Como as fontes de dados são, normalmente, autónomas, distribuídas e heterogéneas, essa actividade lida com o problema de limpar,

[17] Fundamentalmente, Mallach (1994 e 2000), Inmon (1996), Roth (1995), Chaudhuri e Dayal (1997), Dhar e Stein (1997), Wu e Buchmann (1997), Marakas (1999), Chen (2002), Vavouras (2002), Ramakrishnan e Gehrke (2003).

reconciliar e integrar dados das fontes, nas fases de carregamento inicial e de refrescamento periódico dos RDW (Cali et al. 2003: 32-4).

Trata-se do problema de combinar os dados extraídos das fontes e apresentá-los num interface uniforme e global, resolvendo as diferenças de estrutura e de semântica, de modo a providenciar aos utilizadores uma visão reconciliada e unificada dos dados, sem terem que lidar com uma multiplicidade de interfaces, de sistemas e de representações de semântica e de sintaxe (Katchaounov 2003: 1).

A arquitectura do sistema de integração é, normalmente, descrita em termos de dois módulos, os *wrappers* que acedem às fontes, extraem os dados relevantes e apresentam-nos num formato específico e os mediadores que juntam, limpam e combinam os dados produzidos pelos diferentes *wrappers* (ou por outros mediadores), de modo a satisfazer uma necessidade específica de informação (Calvanese et al. 2001: 238).

A integração de fontes de dados ocorre em vários contextos, estando disponíveis sistemas de integração que permitem diferentes tipos de acesso aos dados integrados, bem como abordagens que variam com os diferentes objectivos de integração e assentes em diferentes técnicas e métodos.[18]

Quanto ao acesso aos dados integrados, os sistemas de integração podem ser usados, quer para o acesso só para leitura, quer para o acesso para leitura e escrita (ver Hull 1997: 52, Calvanese et al. 1998a: 280). Os sistemas de integração de acesso para leitura e escrita requerem uma coordenação apertada entre as fontes de dados e entre estas e o interface do sistema de integração, de modo a que as alterações efectuadas nos dados integrados sejam propagadas para a respectiva fonte de dados. Quando a autonomia das fontes de dados se torna um requisito básico que não permite a interferência com as suas operações, os sistemas de integração apenas providenciam o acesso só para leitura, sem permitir a escrita nos dados integrados. O sistema de integração habitualmente implementado nos SDW é de acesso só para leitura.

No que refere às abordagens de integração, elas podem ter por objectivo a integração no nível intencional (dos esquemas das fontes de dados) e no nível

[18] São exemplos de contextos de integração, os sistemas cooperativos e de interoperabilidade, os sistemas de múltiplas bases de dados e os sistemas de informação *Web*. Para uma análise comparativa e crítica das várias abordagens no domínio específico dos SDW, destacam-se os trabalhos de Calvanese et al. (1998a, 1998b e 1999).

extensional (dos dados das fontes), sendo reduzidas as abordagens propostas no domínio específico dos SDW.

Na área da modelação conceptual de bases de dados, as primeiras abordagens trataram o problema da integração no nível intencional. O objectivo é o desenho de um esquema global que reconcilia os diferentes esquemas locais que descrevem os mesmos dados nas múltiplas fontes, de modo a providenciar uma adequada abstracção dos dados das fontes e a produzir uma descrição homogénea de dados relevantes (ver Wiederhold e Elmasri 1979, Navathe et al. 1984, Batini e Lenzerini 1984, Batini et al. 1986).[19]

As abordagens de integração de esquemas constituem um importante contributo na modelação conceptual de bases de dados que integram um conjunto de fontes de dados, concretamente no que refere às metodologias para a especificação do esquema global, incluindo as actividades de análise, comparação, reconciliação e unificação dos modelos intencionais das fontes de dados.[20]

Um aspecto importante na integração de esquemas diz respeito às relações entre as fontes de dados e o esquema global, exigindo a especificação dos mapeamentos entre os elementos dos esquemas locais das fontes de dados e os elementos do esquema global (Calvanese et al. 2003: 30-1). É uma questão fundamental na especificação do processamento de consultas nos dados integrados.

Quanto ao método para especificar as relações entre os elementos dos esquemas das fontes de dados e os elementos do esquema global, distinguem-se duas abordagens: (1) a abordagem *Global-As-View* (GAV), ou baseada no processamento de consultas, em que cada elemento do esquema global é especificado em termos de uma vista sobre os elementos dos esquemas locais das fontes de dados;[21] (2) a abordagem *Local-As-View* (LAV), ou baseada nas fontes de dados, em que cada elemento dos esquemas

[19] Nas metodologias tradicionais, a integração de esquemas era uma actividade única que resultava num esquema global em que todos os dados das fontes são representados uniformemente. Ver a comparação das metodologias de integração de esquemas em Batini et al. (1986: 323-64) e a revisão crítica e comparada de Cali et al (2003), Calvanese et al (2003: 30-6).

[20] No domínio especifico da comparação de esquemas ver, por exemplo, Madhavan et al. (2001 e 2005), Rahm e Bernstein (2001) e os recentes projectos Clio e ToMas dos investigadores da Universidade de Toronto, no sítio http://www.cs.toronto.edu/db/.

[21] Vista é entendida como uma relação derivada, definida em termos de uma ou mais relações base, cujos tuplos são recomputados sempre que a vista é solicitada (Gupta e Mumick 1995: 3 e 1999a: 17-20).

locais das fontes de dados é definido em termos de uma vista sobre os elementos do esquema global (Lenzerini 2002: 235-40).[22]

A abordagem GAV está associada a um método mais focalizado na especificação dos conteúdos das fontes em termos do esquema global e em que a integração de dados não se apoia numa noção explícita de esquemas integrados, assentando mais numa relação algébrica entre as fontes de dados e o DW global, na base do processamento consulta-a-consulta e de um modo *ad hoc* que não atende às necessidades de informação da organização (caso dos projectos WHIPS e Squirrel). A abordagem LAV está associada a um método mais declarativo, cujo objectivo é modelar, numa linguagem apropriada, as relações entre as fontes de dados e o DW global, em termos da informação da organização explicitada num modelo conceptual que representa a visão global dos objectos da organização, independentemente das características físicas e lógicas dos dados nas fontes (caso do projecto DWQ).

Mais recentemente, o esforço tem sido no sentido de se estender a preocupação de integração no nível intencional à integração no nível extensional, tratando-se as fontes de dados em termos dos esquemas locais e dos seus dados, de modo a providenciar uma visão reconciliada e integrada dos dados das fontes (Lenzerini 2002: 233).[23]

Um dos problemas centrais que se coloca na integração, no nível extensional, é o de assegurar que os diferentes objectos que representam as mesmas entidades do mundo real nas diferentes fontes de dados são apresentados como um único objecto na interface do sistema de integração (Calvanese et al. 1998a: 280).

A literatura distingue duas abordagens de integração de dados: (1) a abordagem virtual, assente na utilização de um conjunto de vistas que definem o modo como os dados devem ser combinados para providenciar a visão reconciliada e integrada das fontes de dados; (2) a abordagem materializada, assente na utilização de uma base de dados integrada onde se materializam os dados combinados de acordo com um conjunto

[22] Para uma revisão crítica e comparada destas abordagens, ver Ullman (1997) e Cali et al. 2001 e 2003) e para o domínio específico dos SDW, Calvanese et al. (1998a, 1999 e 2001). Fan propõe uma abordagem que articula as abordagens LAV e GAV, de modo a suportar a evolução dos esquemas locais e global (2005).

[23] Para além dos métodos para capturar os dados das fontes e popular os elementos do esquema global, torna-se fundamental a limpeza dos erros e inconsistências para melhorar a qualidade dos dados (Cali et al. 2001). No capítulo 3, são tratadas estas questões.

de vistas definidas sobre as fontes de dados (Gupta e Mumick 1999a: 17-20).[24] Widom (1995: 25) designa-as por abordagens por solicitação e antecipada, respectivamente, para referir o facto de, no primeiro caso, o processo responsável por extrair, filtrar, combinar e traduzir os dados das várias fontes num formato reconciliado e integrado ser executado no momento em que a consulta é solicitada e, no segundo caso, esse processo ser executado antecipadamente (ver Figura 2.1).

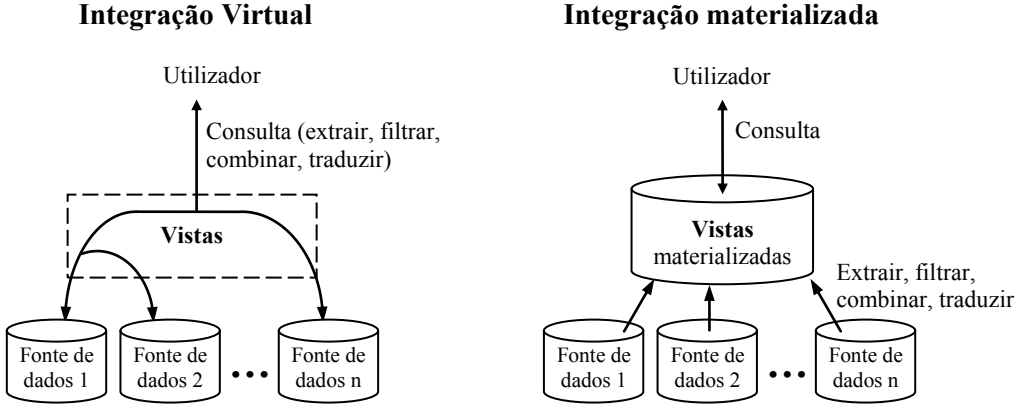


Figura 2.1 Integração virtual e integração materializada

No domínio dos SDW, predomina a abordagem materializada, com raras excepções de propostas de arquitectura para a execução de uma abordagem híbrida, como é o caso do projecto Squirrel (ver Zhou et al. 1995, Hull e Zhou 1996).[25] Apesar de se defender que as vistas (virtuais) são preferíveis quando o número de fontes de dados é elevado, a sua manutenção é frequente e não se prevê o tipo de consultas a serem solicitadas, baixando os custos e o tempo envolvido no seu refrescamento (ver Ashish 2000), o facto é que as fontes de dados raramente guardam o histórico das operações diárias, tornando-se necessário materializar os dados para permitir recriar a sequência de estados correntes dos dados das fontes e disponibilizar informação necessária à análise da evolução histórica dos dados integrados.[26]

Não obstante as abordagens propostas pela investigação e as ferramentas colocadas no mercado pela indústria, o problema da integração das fontes de dados continua na agenda científica das bases de dados.

---

[24] A vista é materializada quando uma cópia dos seus tuplos é armazenada numa base de dados, permitindo, como um *cache*, o rápido acesso aos dados. Quando os tuplos das relações base sofrem alterações, as vistas materializadas ficam desactualizadas (Gupta e Mumick 1995: 3 e 1999a: 17-20).

[25] No capítulo 4 (secção 4.1) apresentam-se os exemplos mais significativos de arquitecturas de manutenção das vistas associadas a estas duas abordagens.

[26] Ver Yang e Widom (1998), Yang (2001: 1-4).

2.3 Componentes da Arquitectura dos SDW

2.3.1 Arquitectura Típica

A integração de fontes de dados com vista à publicação de informação relevante destinada a melhorar as processos de decisão dos gestores e tomada de decisão exige que os SDW sejam implementados de acordo com as características específicas das actividades das organizações e os requisitos dos diferentes utilizadores, tornando-se fundamental a definição da arquitectura e a conexão entre os diferentes componentes.

Na Figura 2.2 apresenta-se uma arquitectura típica que permite identificar os principais componentes: (1) as fontes de dados; (2) as ferramentas de aquisição de dados; (2) os servidores de dados *warehouse*, (3) as ferramentas de acesso aos dados; (4) a gestão do sistema e metadados (Chaudhury e Dayal 1997: 66).[27]

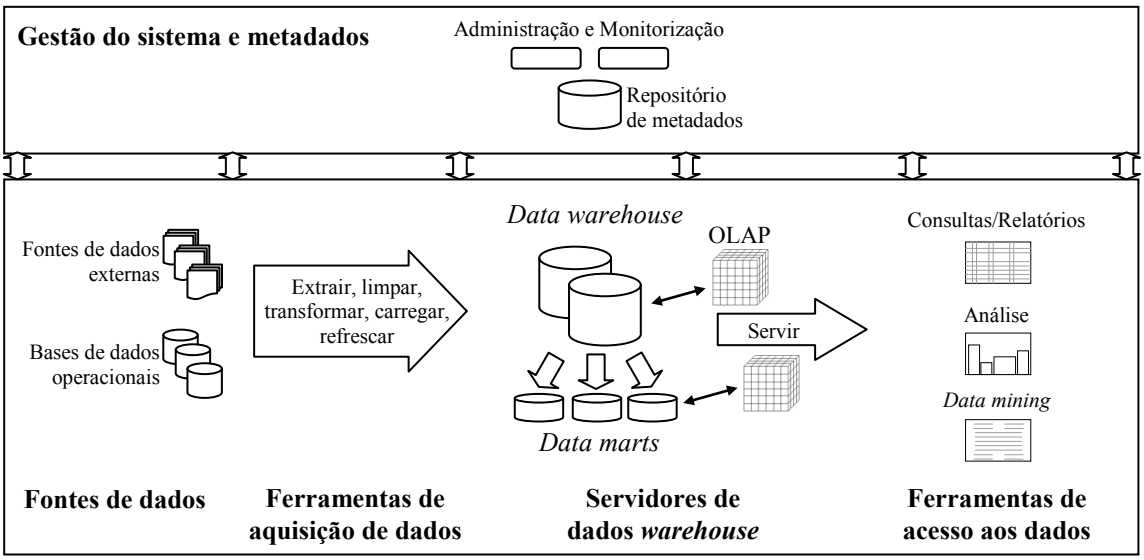


Figura 2.2 Arquitectura típica de SDW

As ferramentas de aquisição de dados constituem a interface entre as fontes de dados e os servidores de dados *warehouse*. São responsáveis por extrair os dados das fontes, limpar as anomalias, reconciliar as diferenças estruturais e semânticas e integrar os dados num formato uniforme e consolidado definido para os dados *warehouse*, calcular os dados derivados e carregar os dados nos RDW. A aquisição de dados divide-

[27] Esta arquitectura típica é apresentada na literatura por vários autores, incluindo Han e Kamber (2001: 66-67), Chen (2002: 53), Ramakrishnan e Gehrke (2003: 870). É designada de arquitectura de três camadas (*three tiers*), em que as fontes de dados e as ferramentas de aquisição de dados constituem a camada 0, o DW e/ou os *data marts* (quase sempre bases de dados relacionais) a camada 1, os servidores de OLAP (ROLAP e/ou MOLAP) a camada 2 e as ferramentas de acesso aos dados a camada 3 (Han e Kamber 2001: 66-7, Eavis 2003: 34-5).

se em duas fases, a do carregamento inicial que assegura a computação inicial dos conteúdos dos RDW e que ocorre num período em que o sistema ainda não está disponível para os utilizadores, e o refrescamento periódico que assegura a sincronização dos dados *warehouse* com os dados das fontes, por forma a reflectir as alterações ocorridas nos dados das fontes, e que ocorre no período de operação do sistema.[28]

Os servidores de dados *warehouse* compreendem os RDW e os Sistemas de Gestão de Bases de Dados (SGBD) que gerem o armazenamento e a manipulação dos dados *warehouse*.[29] Por um lado, as arquitecturas dos RDW podem variar em termos da centralização ou distribuição dos dados *warehouse* (subsecção 2.3.2). Por outro, os SGBD mais utilizados para suportar o processamento OLAP recorrem a abordagens relacionais ou a abordagens multidimensionais para armazenar os dados *warehouse* (subsecção 2.3.3).

As ferramentas de acesso aos dados permitem aceder, apresentar e interagir com os dados publicados, oferecendo funcionalidades de consultas simples e visualização de resultados, de emissão de relatórios, de análise que varia desde o simples cálculo de médias até cálculos complexos de aplicação de métodos estatísticos avançados e apresentação de dados em diferentes formatos, incluindo matrizes, gráficos 3-D, etc. As ferramentas de OLAP e de *data mining* são as mais popularizadas (subsecção 2.3.4). Os dados publicados podem também servir outros sistemas e aplicações externas, como os sistemas de informação para executivos, os sistemas de planeamento de recursos empresariais, as aplicações de gestão de risco financeiro ou as aplicações de gestão de relações com os clientes (Inmon 1996: 243-60 e Inmon et al. 2001: 202-9).[30]

E por último, o componente de gestão do sistema que usualmente compreende ferramentas disponibilizadas no mercado para administrar e monitorizar aspectos concretos do sistema, mas em que os metadados têm um papel fundamental em todo o ciclo de vida dos SDW (subsecção 2.3.5).

---

[28] O processo de refrescamento é pormenorizado no capítulo 3.

[29] O SGBD recorre a um conjunto de técnicas para gerir grandes volumes de dados, processar as consultas e providenciar um meio programático de acesso aos dados (por exemplo a SQL). Informação detalhada sobre os SGBD pode ser encontrada em Ramakrishnan e Gehrke (2003), Silberschatz et al. (2006).

[30] Alguns destes sistemas e aplicações externas podem assumir o papel de fonte de dados e o papel de cliente dos SDW.

2.3.2 Repositórios de *Data Warehouse*

Os dados *warehouse* podem estar centralizados num único repositório ou distribuídos por vários repositórios, podem dizer respeito a um domínio local ou a um domínio global das actividades da organização e podem ser estruturados em camadas com diferentes níveis de granularidade ou de acordo com o momento a que se referem, de modo a responder às características das actividades das organizações e aos diferentes requisitos dos utilizadores.[31]

Na arquitectura centralizada existe apenas um DW global que armazena (nos níveis lógico e físico) todos os dados necessários à análise das actividades da organização. Enquanto que na arquitectura distribuída coexistem múltiplos RDW, num mesmo lugar ou em vários lugares, e os dados são fisicamente distribuídos, podendo optar-se por consolidar os dados dos múltiplos RDW no nível lógico - arquitectura federada - ou por consolidar os dados fisicamente num DW global e redistribui-los por sucessivas camadas de *data marts* que armazenam cópias ou sumários das camada anteriores - arquitectura de camadas (ver Figura 2.3).

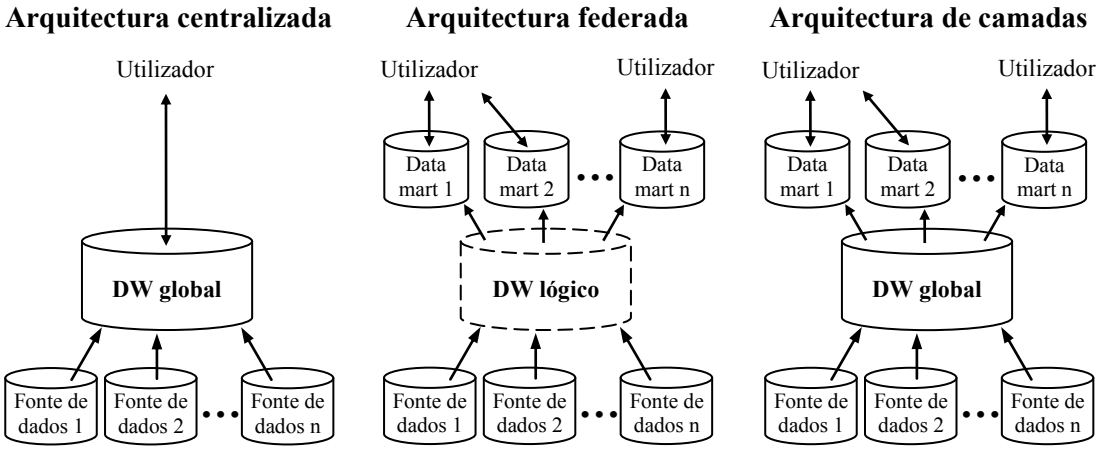


Figura 2.3 Arquitectura centralizada e arquitecturas distribuídas

As soluções centralizadas são úteis nas organizações em que o ambiente operacional é centralizado e têm a vantagem de utilizar um único RDW (DW global), facilitando a construção e manutenção do sistema e permitindo o acesso uniforme a um único modelo de dados, mas perdem em performance, quando comparadas com as soluções distribuídas. De facto, as arquitecturas distribuídas, mais adequadas a

[31] A granularidade diz respeito ao nível de detalhe ou sumarização dos dados, sendo tanto mais elevado, quanto menos detalhado ou mais sumariado se apresentarem os dados *warehouse* (Inmon 1996 : 45-6).

organizações com ambientes operacionais distribuídos, permitem uma resposta mais rápida, porque os dados estão distribuídos por múltiplas máquinas e localizados mais próximos das aplicações cliente, reduzem o volume de dados a serem pesquisados e facilitam a extensão e reestruturação dos SDW (Jarke e Quix 2003: 10-2).[32]

As arquitecturas distribuídas colocam a questão dos *data marts* e a sua relação com o DW global.

Do ponto de vista de Kimball et al., o *data mart* usa um modelo desnormalizado (em *constelação*) e é entendido como um subconjunto lógico do DW global que suporta a análise (OLAP) de um dado assunto que pode ser de interesse para vários departamentos e utilizadores, mas que pode incluir dados detalhados e dados históricos (1998: 18, 27-8).

Na perspectiva de Inmon et al., o DW global usa um modelo normalizado de base de dados, centraliza todos os assuntos e armazena dados detalhados e dados históricos, enquanto que o *data mart* usa um modelo desnormalizado (em *estrela*), é criado a partir de dados sumariados do DW global (com pequenas porções de dados detalhados) para representar um assunto de interesse departamental, pode constituir-se como um servidor de OLAP e funciona como uma interface entre o DW global e a camada de acesso aos dados (2001: 102-3 e 112-5). [33]

Desta oposição decorre ainda a distinção entre os *data marts* dependentes que publicam os dados extraídos de um DW global e os *data marts* independentes que publicam os dados extraídos directamente das fontes de dados (ver Chen 2002: 56-7). Note-se que os *data marts* independentes podem extrair dados das mesmas fontes, não exigem processos de ETC comuns e os metadados são distribuídos, tornando-se complexa a sua manutenção (ver Figura 2.4).

[32] As arquitecturas distribuídas podem servir um domínio local ou uma intersecção de múltiplos domínios de actividades, satisfazendo as necessidades de partilha de dados nas tomadas de decisão, a nível local ou global (Inmon et al. 2001: 267-90).

[33] Ver a arquitectura *Bus* de Kimball et al. (1998) e a arquitectura *Corporate Information Factory* de Inmon et al. (2001).

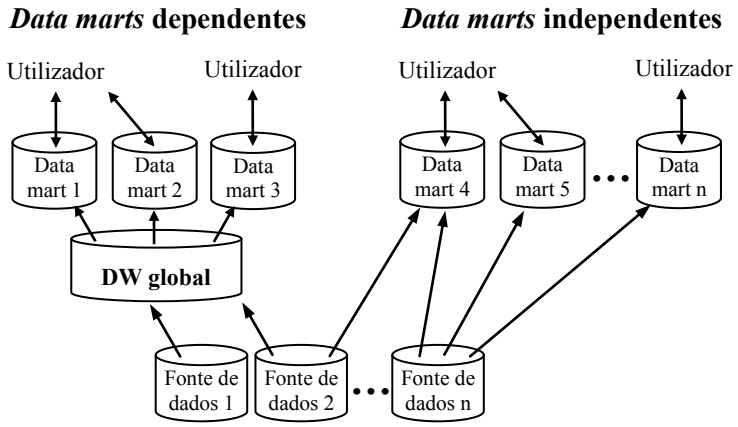


Figura 2.4 *Data marts* dependentes e *data marts* independentes

O facto é que as arquitecturas evoluíram e as soluções conjugam os DW globais e os *data marts* para organizar e armazenar os dados *warehouse*, de acordo com os interesses de informação específicos de uma organização. Por exemplo, numa hierarquia de camadas de dados com diferentes níveis de granularidade e/ou de acordo com o momento a que se referem (ver Figura 2.5). Desde as fontes de dados que contém os dados no mais alto nível de detalhe até às estruturas de dados especializadas que contêm dados muito sumariados. E desde os dados mais antigos, usualmente armazenados em estruturas alternativas que apresentam custos inferiores e uma menor performance de acesso, até aos dados mais recentes que reflectem o estado corrente das fontes de dados (Inmon 1996: 37-38).

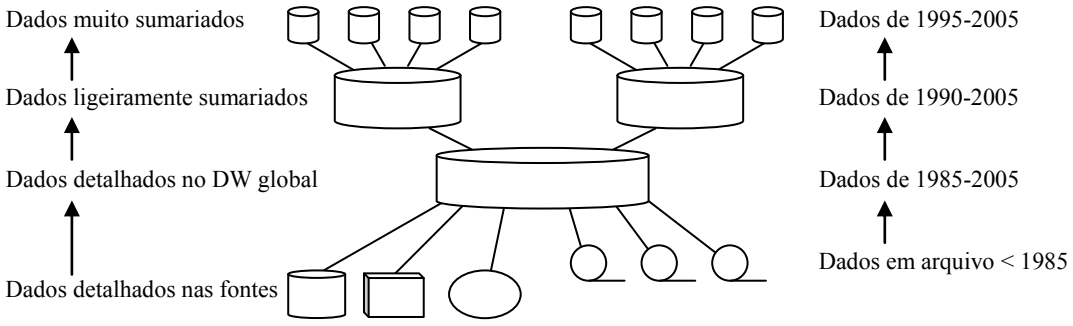


Figura 2.5 Dados *warehouse*, por nível de detalhe e idade

A necessidade das organizações acederem a dados actuais, integrados e consolidados para a tomada de decisões operacionais e tácticas potenciou a utilização do ODS, sobretudo nas organizações com um elevado número de processamentos de transacções online. O ODS é um repositório de dados orientados a assuntos, integrados, voláteis, correntes e detalhados e que para autores como Inmon deve ser utilizado no

ambiente operacional como um ponto intermédio de integração de um conjunto de fontes de dados, providenciando o acesso a dados actuais e consolidados (2001: 79-80). Todavia, para outros como Kimball et al. deve fazer parte do ambiente *warehouse*, para possibilitar aos decisores o acesso a dados operacionais mais actuais e detalhados (1998: 19-20 e 340).

Alguns autores têm vindo a defender que as necessidades satisfeitas pelos ODS no ambiente *warehouse* podem ser supridas pelos recursos oferecidos pelos sistemas de *real-time data warehouses*, nomeadamente no que trata ao suporte à tomada de decisões operacionais e tácticas, providenciando aos gestores informação relevante e em tempo real para que possam decidir atempadamente (Kimball e Caserta 2004: 421-60).[34]

Esta abordagem alternativa rompe com o pressuposto que os RDW são refrescados periodicamente e em diferido para providenciar dados que possibilitem uma tomada de decisão atempada. Algumas propostas de *real-time data warehouses* envolvem o recurso a técnicas de particionamento para armazenar, em separado, os dados correntes (refrescados com muita frequência) e os dados antigos (refrescados com pouca frequência) e/ou às capacidades activas das fontes de dados para detectar e notificar de imediato as alterações ocorridas nos seus conteúdos (Buckner et al 2002: 318, Nguyen e Tjoa 2003, Kühn 2003, Kimball e Caserta 2004: 421-60).

2.3.3 Abordagens de Armazenamento para o Processamento OLAP

Para suportar o processamento OLAP é necessário reproduzir uma visão conceptual multidimensional dos dados *warehouse* (Codd et al. 1993: 4 e 12). No que trata aos modelos lógicos e físicos que reproduzem a visão conceptual multidimensional, o debate tem-se situado em torno de duas abordagens genéricas de armazenamento: (1) a abordagem *Multidimensional Online Analytical Processing* e (2) a abordagem *Relational Online Analytical Processing* (ver Vassilidadis e Sellis 1999:

[34] Estes sistemas são também designados por *zero-delay data warehouses* e *zero-latency data warehouses* (Buckner et al 2002, Kühn 2003) ou *real-time ETL* (Kimball e Caserta 2004: 421-60). A necessidade de agilizar as decisões operacionais e tácticas, de acordo com a estratégia global das organizações está na agenda dos SDW (Golfarelli et al. 2004).

64, Han e Kamber 2001: 67, Pedersen e Jensen 2001: 43-4, Chen 2002: 66-7, Franconi et al. 2003: 87-105).[35]

Na abordagem *Multidimensional Online Analytical Processing* (MOLAP), os dados são armazenados em matrizes que reproduzem a visão conceptual multidimensional dos objectos de análise, no nível físico. Os elementos das matrizes contêm os valores das métricas do objecto de análise e a posição de um elemento é determinada pela combinação dos valores das dimensões que caracterizam o objecto de análise.

Na Figura 2.6 o objecto de análise *venda* é caracterizado pelas dimensões *região*, *produto* e *data*.

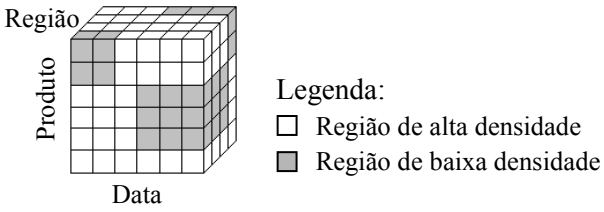


Figura 2.6 Armazenamento MOLAP (matriz de 3 dimensões)

Dada a natureza multidimensional do modelo físico, as operações de OLAP podem ser executadas nativamente por um motor apropriado de processamento de consultas (motor MOLAP), tirando proveito da natureza posicional das matrizes para indexar os elementos da matriz e para simplificar as actividades de análise mais comuns e evitando operações potencialmente dispendiosas, como o processamento de consulta cruzada de duas ou mais tabelas interligadas (*joins*), de consultas compostas por várias sub-consultas (*nested queries*) e a união dos dados residentes em múltiplas tabelas (*unions*), habitualmente associadas ao processamento de consultas em modelos relacionais. Por outro lado, a utilização de matrizes que mantêm todas as combinações possíveis das dimensões, nos diferentes níveis de agregação, possibilita uma elevada performance no processamento de consultas aos dados, antecipando e restringindo as diferentes maneiras como os dados podem ser acedidos (Kimball et al. 1998:427, Franconi et al. 2003: 95-6).

[35] As abordagens MOLAP e ROLAP são as que têm recebido uma maior atenção por parte da comunidade cientifica e da indústria. Contudo, existem abordagens assentes noutros modelos, como o *Object-Oriented* OLAP (Buzydlowski et al. 1998, Gopalkrishnan et al. 1999). Estão disponíveis arquitecturas físicas associadas à duas abordagens (Wu e Buchmann 1997: 69-70, Kimball et al. 1998: 407-8), comparações das suas performances (Colliat 1996, Kimball et al. 1998: 425-8) e suas facilidades de utilização (Gorla 2003).

Uma das questões centrais que tem vindo ser estudada está associada ao modo como se consegue garantir um armazenamento eficiente dos dados quando se utiliza um modelo de elevada dimensionalidade. Verifica-se que a matriz que reproduz um modelo de elevada dimensionalidade, depois de armazenar todas as combinações nos diferentes níveis de agregação, apresenta um elevado número de elementos vazios, isto é, muitas das posições da matriz apontam para métricas sem qualquer valor atribuído - matriz de baixa densidade (ver Figura 2.6). As soluções mais comuns têm sido as heurísticas para identificar as regiões de baixa densidade nas matrizes, as técnicas de compressão de regiões de baixa densidade e as técnicas híbridas de indexação que combinam o armazenamento das regiões de alta densidade em bases de dados multidimensionais com o armazenamento das regiões de baixa densidade em bases de dados relacionais (Chaudhury et al. 2001: 51, Kaser e Lemire 2003:1-2, Deveaux 2004: 1-2).[36]

Nesta abordagem, contudo, regista-se a inexistência de um modelo de dados lógico multidimensional standard que permita abstrair das características físicas das ferramentas MOLAP de diferentes fornecedores e possibilite o desenho lógico independentemente da plataforma utilizada (Franconi et al. 2003: 96).

Na abordagem *Relational Online Analytical Processing* (ROLAP), utilizam-se modelos lógicos relacionais estendidos assentes em construções muito próximas dos conceitos multidimensionais para construir uma camada semântica que providencie as funcionalidades OLAP sobre os dados armazenados em bases de dados relacionais, traduzindo as actividades de análise para a linguagem nativa da base de dados relacional (Colliat 1996: 65 e Franconi et al. 2003: 92).

Esta abordagem tem a vantagem de utilizar um modelo lógico que atingiu um avançado nível de maturidade, que tem sido amplamente aceite pela indústria e que se popularizou entre os práticos no desenvolvimento de bases de dados operacionais e aplicações associadas, como as aplicações de OLTP. Todavia, as técnicas de modelação relacional mostram-se pouco apropriadas para as aplicações analíticas que exigem uma grande flexibilidade no modo com se combinam e visualizam os dados, sacrificando, muitas vezes, o processamento de consultas sofisticadas habitualmente associadas às actividades de análise (Codd et al. 1993: 1-4 e Franconi et al. 2003: 92-3).[37]

[36]Ver exemplos destas técnicas em Cheung et al. (2001), Kaser e Lemire (2003).

[37] As técnicas de modelação relacionais têm se mostrado úteis no desenvolvimento de aplicações de processamento operacional, transaccional e complexo. Utilizam-se as formas normais para eliminar a redundância de dados e optimizar o processamento das transacções que alteram os dados (inserções,

No desenvolvimento de aplicações de processamento OLAP são utilizadas, por isso, as técnicas de modelação dimensional que recorrem a construções básicas assentes em factos e dimensões para constituir estruturas mais apropriadas às características das aplicações analíticas, particularmente os esquemas em estrela ou em floco-de-neve (Wu e Buchmann 1997: 62-4 e Kimball et al. 1998: 140-52).

O facto (ou tabela de facto) é o elemento central de um modelo dimensional e representa o objecto de análise, contendo um conjunto de métricas e um conjunto de apontadores (ou chaves estrangeiras) para cada uma das dimensões (ou tabelas de dimensão), cada uma delas descrevendo uma dada perspectiva do objecto de análise, através de um conjunto de atributos, normalmente textuais, que podem ser organizados segundo diferentes caminhos de consolidação (ou hierarquias). Os apontadores do facto definem as coordenadas dimensionais do objecto de análise, reproduzindo a visão multidimensional (Wu e Buchmann 1997: 62-4 e Kimball et al. 1998: 140-52).

O esquema em estrela é constituído por um facto central e múltiplas dimensões associadas. Para eliminar a redundância de dados e minimizar os custos de armazenamento associados pode-se normalizar algumas dimensões, criando um esquema em floco-de-neve. Diz-se que uma dimensão é floco-de-neve quando os atributos que apresentam um menor número de valores diferentes - atributos de menor cardinalidade - são removidos para novas dimensões criadas em separado e que, por sua vez, ficam interligadas à dimensão original, através de um apontador artificial (Kimball et al. 1998: 170).

No esquema em estrela ilustrado na Figura 2.7 (lado esquerdo), o facto *venda* contém a métrica *valor da venda* e o conjunto de apontadores *chave produto*, *chave data*, *chave região* e *chave cliente* para as dimensões *produto*, *data*, *região* e *cliente*, respectivamente. As dimensões contêm um apontador para o facto *venda* e um conjunto de atributos textuais, por exemplo, a dimensão *região* contém o apontador *chave* e os atributos *distrito*, *concelho*, *localidade*. O modelo *estrela* não representa explicitamente as hierarquias das dimensões, mas pode ser mais apropriado para navegar nas dimensões.

No esquema em floco-de-neve também ilustrado na Figura 2.7 (lado direito), por exemplo, a dimensão *produto* é uma dimensão floco-de-neve, pois os atributos de

---

actualizações e eliminações). Para uma descrição pormenorizada destas técnicas ver, por exemplo, Ramakrishnan e Gehrke (2003: 605-40) e Silberschatz et al (2006: 263-305).

menor cardinalidade *categoria* e *família* foram removidos para as dimensões *categoria* e *família*, que ficam interligadas à dimensão *produto*, através dos apontadores artificiais *chave categoria* e *chave família*, respectivamente. As hierarquias das dimensões são explicitamente representadas, através da normalização das tabelas de dimensões, o que oferece vantagens na sua manutenção.

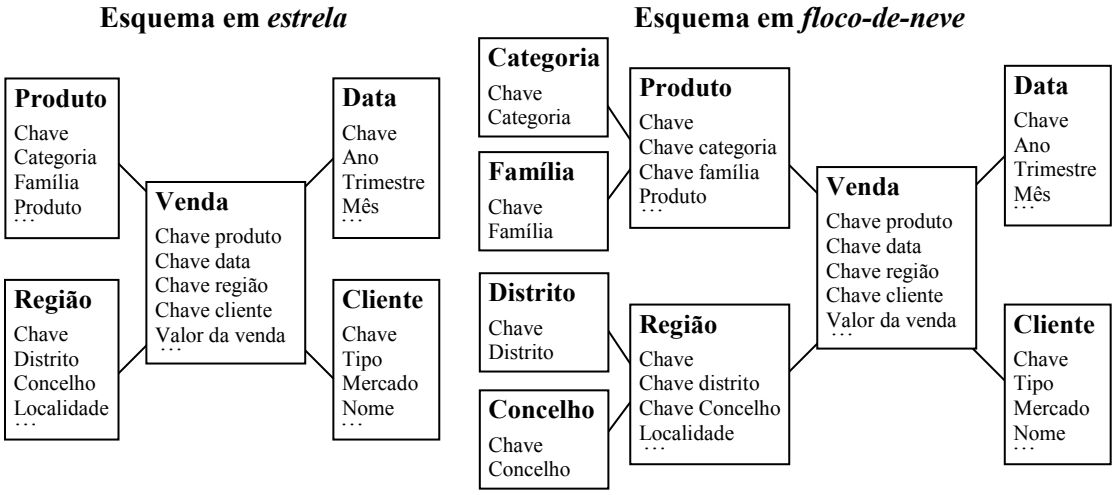


Figura 2.7 Esquemas em estrela e floco-de-neve (ROLAP)

O esquema em constelação é a alternativa proposta por Kimball et al. baseada no conceito de dimensões conformes que permitem que as hierarquias de dimensões partilhadas por diferentes factos figurem em dois ou mais esquemas em estrela com diferentes granularidades, de modo a suportar analises cruzadas de diferentes factos através da navegação numa dimensão comum (1998: 153-64).

O modelo dimensional rompe com a teoria da normalização definida no âmbito do bem sucedido modelo relacional para tirar proveito das vantagens de realizar a visão multidimensional, de modo a suportar as actividades de análise dos dados. Primeiro, a simplicidade e previsibilidade do modelo dimensional permitem reduzir a complexidade das ferramentas de acesso aos dados, fornecendo um *framework* standard que torna as interfaces mais intuitivas e favorecendo o processamento eficiente das consultas aos dados. Segundo, a simetria das dimensões face aos factos facilita as ferramentas de acesso aos dados a lidar com o processamento de consultas *ad-hoc*, com alterações inesperadas no comportamento dos utilizadores e nos padrões de consultas. Terceiro, a capacidade do modelo dimensional lidar com as características específicas da análise dos dados, oferecendo abordagens standard para a resolução de situações comuns nas organizações. Quarto, a vocação do modelo dimensional para suportar a definição de diferentes caminhos de consolidação (hierarquias) que podem ser explorados através de

operações de navegação e agregação. Finalmente, a escalabilidade dos modelos dimensionais que se traduz na facilidade de alteração do modelo, à medida que os requisitos dos utilizadores evoluem no tempo (Kimball et al. 1998: 47-50).

A abordagem ROLAP tem sido uma solução muito adoptada para o armazenamento de dados multidimensionais, tendo-se verificado melhorias significativas, concretamente nas extensões criadas para dar resposta às características específicas do processamento OLAP, incluindo o suporte ao desenho de modelos dimensionais, as técnicas de optimização do processamento de consultas de tabelas de factos com múltiplas dimensões associadas (*joins* em estrela) e respectivas técnicas de indexação de dados *join* e *bitmapped join* e optimizadores para reduzir os custos da execução das operações nativas das bases de dados relacionais (Kimball et al. 1998: 426, Ramakrishnan e Gehrke 2003: 866 e 869).

2.3.4 Ferramentas de OLAP e de Data Mining

Nos últimos anos, as ferramentas de OLAP e de *data mining* têm vindo a ser utilizadas nos SDW, tendo-se revelado úteis no apoio às actividades de análise de dados e tomada de decisão (Vavouras 2002: 13, Chen 2002: 6, Kimball 1998: 401-5).

Desde o já clássico Codd et al. (1993) que a funcionalidade oferecida pelo processamento OLAP tem tido um lugar de destaque entre as ferramentas de análise dos dados nos SDW. Possibilita aos utilizadores uma navegação simples e intuitiva sobre os dados da organização e ao longo de diferentes cenários especulativos do tipo hipotético (e se?) ou justificativo (porquê?), num contexto histórico e de múltiplas dimensões e caminhos de consolidação, e promove uma análise dinâmica e dirigida pelas motivações e necessidades específicas dos utilizadores (Pedersen e Jensen 2001: 40-47, Kimball e Caserta 2004: 247-8).[38]

Os aspectos estáticos do processamento OLAP compreendem um objecto de análise e um conjunto de variáveis, cada uma delas representando uma dimensão que caracteriza o domínio deste objecto. O objecto de análise pode ser definido como uma função das correspondentes variáveis, por exemplo, *v=f(p, r, d)*, em que *v* refere o objecto de análise *venda* e *p*, *r* e *d* representam as dimensões *produto*, *região* e *data* que o caracterizam, respectivamente. Podem ser definidas várias métricas para quantificar um dado objecto de análise e múltiplos caminhos de consolidação (hierarquias) em cada

[38] Ver as doze regras para avaliar as ferramentas de OLAP propostas por Codd et al. (1993).

uma das dimensões. A combinação dos valores das várias dimensões determinam univocamente o valor das métricas (Chaudhuri e Dayal 1997: 68 e Wu e Buchmann 1997: 62-4).

Por outro lado, os aspectos dinâmicos do processamento OLAP dizem respeito às várias actividades de análise, incluindo o *roll-up*, o *drill-down*, o *pivot*, o *slice,* o *dice* e o *rank* definidas na Tabela 2.2 (Pedersen e Jensen 2001: 43). As operações primitivas que tornam possíveis estas actividades incluem a comparação, a ordenação e a consolidação de dados. Destaca-se a importância da operação de consolidação dos dados, responsável pelo processo de agregar os valores detalhados do objecto de análise (métricas) em blocos únicos de informação sumariada, de acordo com um nível na hierarquia de uma dada dimensão (Wu e Buchmann 1997: 62).

Slice* e *Dice
São actividades de análise que permitem a selecção e a projecção de um conjunto de dados que satisfazem uma condição numa dada dimensão (*slice*) ou em duas ou mais dimensões do cubo (*dice*) para reduzir a dimensionalidade do cubo
Drill
São actividades de análise prospectiva que permitem usar as hierarquias das dimensões. O *drill-down* ou *roll-down* permite navegar dos níveis de dados mais sumariados para os níveis de dados mais detalhados dentro de uma dimensão ou introduzir uma nova dimensão; o *drill-up* ou *roll-up* permite sumariar os dados, navegando-se dos níveis de dados mais detalhados para os níveis de dados mais sumariados dentro de uma dimensão ou reduzindo-se uma dimensão; o *drill-across* permite análises transversais que envolvem mais do que uma tabela de factos: o *drill-through* permite descer no nível de detalhe até às fontes de dados
Rank Top n* e *Botton n
É uma actividade de análise que permite usar os dados que se encontram na base (*bottom n*) ou no topo (*top n*) de uma ordem específica
Rotate* ou *Pivot
É uma actividade de análise que permite a rotação dos eixos de visualização dos dados, usando um nível de dimensão como variável independente. Permite a representação dos dados em 3D numa série de planos 2D.

Tabela 2.2 Actividades de análise OLAP mais comuns

Na sessão de OLAP ilustrada na Figura 2.8, analisa-se a métrica *quantidade vendida* do objecto de análise *venda*, de acordo com as dimensões *data*, *produto* e *região*. A dimensão *data* é organizada segundo a hierarquia *ano-trimestre-mês*. A *quantidade vendida* de 50 unidades é determinada pela combinação dos valores das dimensões *produto* "Software", *região* "Lisboa" e *data* "Janeiro de 2005".

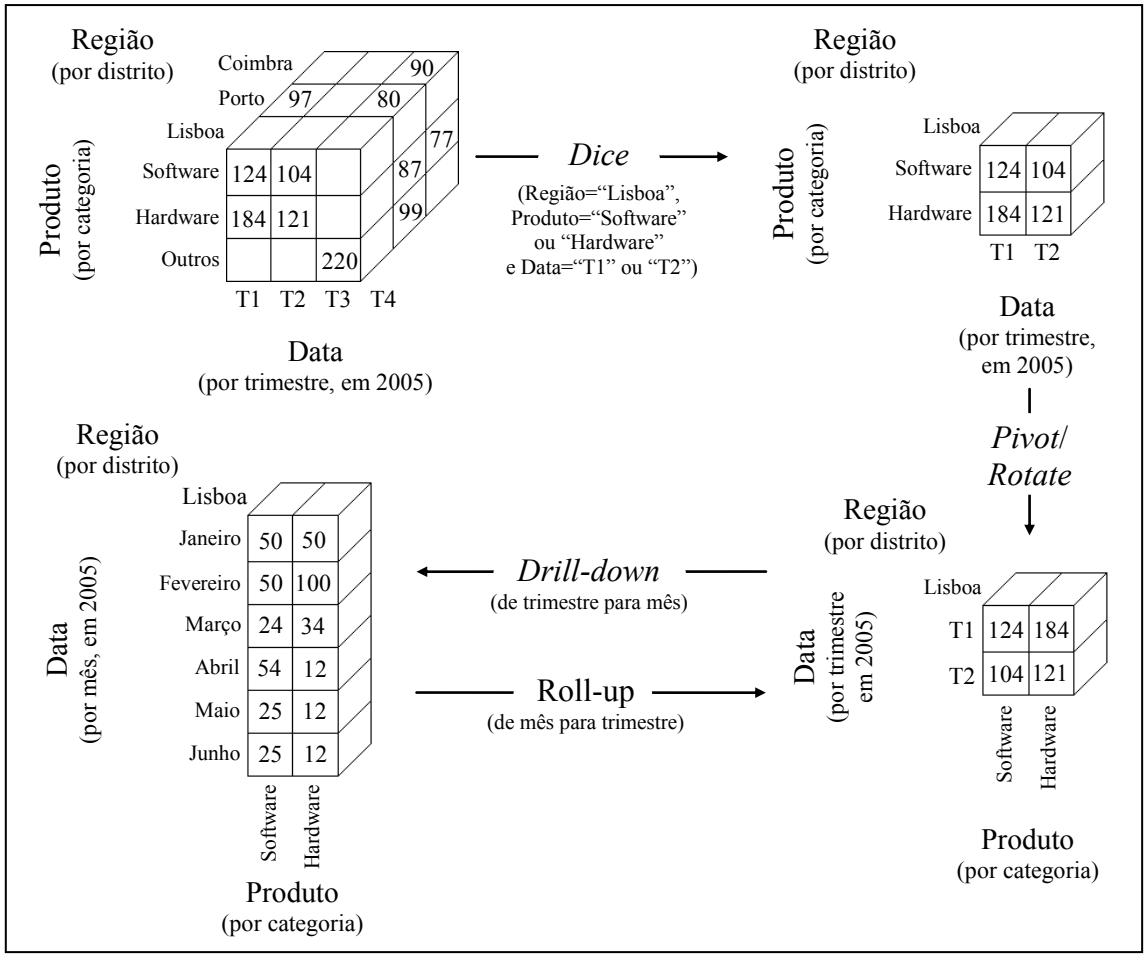


Figura 2.8 Exemplo de sessão OLAP

Por seu lado, a descoberta de conhecimento em bases de dados e o *data mining* são áreas de investigação que têm vindo a receber a atenção de muitos autores nos últimos anos, reunindo as práticas e saberes de várias disciplinas, incluindo a estatística, a tecnologia em bases de dados, a inteligência artificial, a visualização e a ciência da computação (Fayyad 1998: 39, Piatetsky-Shapiro 2000: 59-61, Hand et al. 2001: 4 e Fayyad et al. 2003 191-6).[39]

Estas áreas de investigação suscitam dois entendimentos. A descoberta de conhecimento em bases de dados como o processo não trivial de identificar estruturas válidas (padrões ou modelos), previamente desconhecidas, potencialmente úteis e com significado, a partir dos dados e o *data mining*, como a aplicação de algoritmos de análise de dados e de descoberta de padrões que, sobre limitações de eficiência computacional aceitáveis, enumera um conjunto de estruturas (modelos ou padrões)

[39] Kimball et al. (1998: 401) e Chen (2002: 6) referem que o aparecimento dos SDW promoveu a oferta de ferramentas industriais de *data mining*. Fayad admite mesmo o crescimento do *data mining* é devido à procura e não ao sucesso da área (Fayyad et al. 2003: 192).

identificadas nos dados, constituindo um importante passo do processo de descoberta de conhecimento em bases de dados (Fayyad et al. 1996: 40-2).[40]

A Figura 2.9 ilustra o processo de descoberta de conhecimento em bases de dados nos seguintes passos: (1) a selecção dos dados (amostra); (2) a limpeza e pré-processamento dos dados seleccionados; (3) a transformação dos dados; (4) a identificação e enumeração de estruturas (*data mining*); e (5) a avaliação e possível interpretação das estruturas identificadas para determinar quais as estruturas que podem ser consideradas como conhecimento. O processo é conduzido pelo utilizador, podendo envolver múltiplos ciclos entre quaisquer dos seus passos (Fayyad et al. 1996: 40-2).

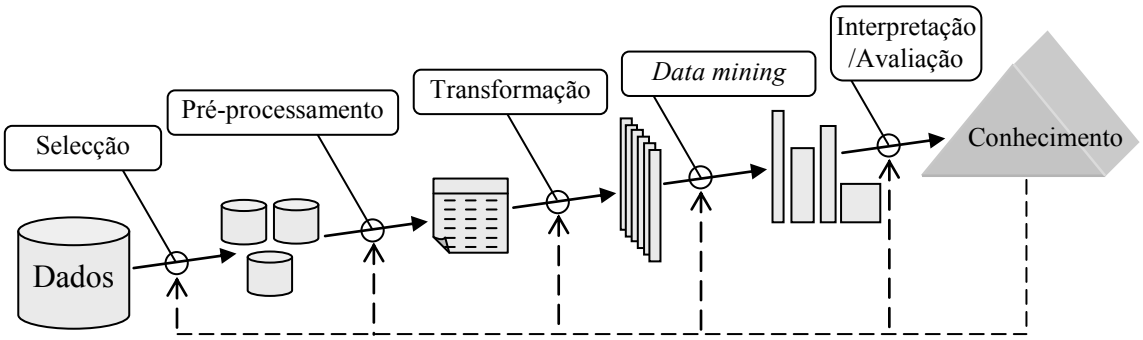


Figura 2.9 Processo de descoberta de conhecimento em bases de dados

Do processo de descoberta de conhecimento em bases de dados podem resultar diferentes estruturas, incluindo equações lineares, regras, segmentos (*clusters*), gráficos e estruturas em árvore, sendo usual distinguirem-se as estruturas (globais) que são aplicáveis a todo o conjunto de dados, em todo o espaço de medida (modelo) e as estruturas (locais) que são aplicáveis apenas a um subconjunto dos dados ou uma parte do espaço de medida (padrão) (Hand et al. 2001: 1, 9-11).

As iniciativas de descoberta de conhecimento em bases de dados podem ser desenvolvidas com diferentes motivações e objectivos, permitindo-se classificá-las em duas grandes categorias: (1) a previsão, recorrendo às técnicas que permitem estimar os valores (desconhecidos ou futuros) de variáveis de interesse a partir dos valores (conhecidos) de outras variáveis, distinguindo-se a descoberta de valores de variáveis organizadas em categorias - classificação - e de variáveis quantitativas (contínuas) -

[40] Os termos descoberta de conhecimento e *data mining* são muitas vezes utilizados indiscriminadamente, embora haja um consenso no que trata a uma clara separação entre o processo geral, incluindo a selecção dos dados (amostra) e a limpeza/pré-processamento dos dados e a avaliação e possível interpretação dos padrões ou modelos identificados (processo de descoberta de conhecimento) e a aplicação dos algoritmos sobre os dados (*data mining*) (Fayyad 1998: 41, Piatetsky-Shapiro 2000: 59).

regressão -; (2) a descrição, recorrendo às técnicas que permitem descobrir padrões ou regras de associação interessantes nos dados e criar modelos descritivos dos dados ou dos processos que geraram esses dados (Fayyad et al. 1996: 44 e Han e Kamber 2001: 179).

O *data mining* e a descoberta de conhecimento em bases de dados têm-se mostrado úteis em vários domínios de aplicação, como por exemplo, no área do marketing, incluindo a descoberta de variações do volume de vendas, padrões de consumo, fidelidade e comportamento do público, relações entre as variáveis do cesto de compras; na área financeira, designadamente a descoberta dos factores que podem influenciar o cumprimento dos créditos, modelos previsionais que revelem as tendências do mercado das diferentes bolsas de investimento; na indústria das telecomunicações, como por exemplo a descoberta de padrões de consumo dos telefones móveis e lançamento de novos tarifários, padrões que indiciem a utilização fraudulenta de telefones móveis, padrões de regularidade e irregularidade nos sistemas de gestão da rede (Barchman et al. 1996: 44-7, Hand et al. 2001: 11-5 e Han e Kamber 2001: 451-7).

2.3.5 Papel dos Metadados

Hoje, quase todos os componentes das tecnologias de informação contêm metadados que suportam as suas funcionalidades. Nos últimos anos, os metadados tornaram-se um domínio de extrema importância em todas as fases do ciclo de vida dos sistemas de informação, sendo fundamentais na gestão do sistema, particularmente no que refere à publicação, acesso e exploração dos dados (Friedrich 2005: 799-80).

De facto, os *terabytes* e *petabytes* de dados geridos pelos sistemas de informação têm feito crescer o mercado de vendedores de vários tipos de tecnologia de gestão de metadados (ver Kim 2005), bem como o esforço, também de pesquisa, no sentido de soluções que possam facilitar a partilha e reutilização de metadados entre utilizadores e componentes que operam entre si nos sistemas de informação (Vaduva et al. 2001: 85). Têm sido também fundamentais as iniciativas de estandardização, como por exemplo o *Common Warehouse Metamodel*, um standard de interoperabilidade do *Object Management Group* que define uma linguagem comum e mecanismos de partilha para especificar metadados (OMG 2001).

A complexidade e extenso uso de metadados têm tornado difícil a definição precisa desta noção muito vulgarizada na investigação e na prática como os *dados sobre os dados*. No domínio da modelação e desenho de bases de dados, Blaha e Premerlani

definem metadados no contexto de uma hierarquia de quatro níveis de modelação de dados, em que cada nível é o nível *meta* do nível inferior, ou seja, contém as construções ou a linguagem de modelação do nível inferior, logo representa dados de um nível superior de abstracção: (1) o nível 0, dos *dados* propriamente ditos sobre os conceitos e relações que definem uma dada aplicação; (2) o nível do *modelo* entendido como uma abstracção dos *dados*; (3) o nível dos *metadados* entendidos como uma abstracção do *modelo*; (4) o nível superior do *metamodelo* entendido como uma abstracção dos *metadados* (1998: 75-7).

Para ajudar a esclarecer este domínio complexo torna-se útil a classificação de metadados proposta de Staudt et al, bem como o papel que desempenham e os objectivos que permitem alcançar nos SDW (1999a: 37-62 e 1999b).

Os metadados podem oferecer uma abstracção conceptual dos dados numa linguagem próxima da linguagem natural, ou uma abstracção lógica numa linguagem matemática, ou uma abstracção física em SQL (Staudt et al 1999a: 43-4). Podem ainda desempenhar um papel passivo, providenciando documentação sobre a estrutura, o processo de desenvolvimento e a utilização dos SDW para apoiar os *stakeholders* nas suas tarefas, ou um papel activo, especificando as regras que conduzem a execução dos processos; ou ainda um papel semi-activo, oferecendo dados que descrevem aspectos estáticos que são lidos por outros componentes de software durante a sua execução (Staudt et al 1999b: 5-6).

Por outro lado, os metadados permitem, do ponto de vista técnico, minimizar o esforço de desenvolvimento e gestão do sistema, designadamente no automatismo dos processos de carregamento inicial e de refrescamento dos RDW, na integração das fontes de dados e dos dados, no reforço dos mecanismos de segurança do sistema, na análise e desenho de novas aplicações, na maior flexibilidade do sistema e reutilização dos módulos de software. Do ponto de vista do utilizador, os metadados permitem melhorar a informação publicada, no que refere à qualidade, acesso, utilização e análise (Staudt et al 1999b: 6-7).

O facto dos SDW reflectirem o modelo da organização torna a gestão de metadados um aspecto central na arquitectura. Os SDW têm que gerir vários tipos de metadados, desde os que permitem entender e pôr o sistema a funcionar, os que descrevem todos os repositórios de dados e seus conteúdos, os mapeamentos, os processos, os que definem o acesso à informação (Chaudhury e Dayal 1997: 73).

A investigação tem divulgado *frameworks* de desenho e desenvolvimento que propõem diferentes sistemas de gestão de metadados: (1) as abordagens assentes num repositório de metadados que garante a partilha e reutilização de metadados por todos os componentes e que serve, fundamentalmente, o administrador do sistema e o designer, mas que pode colidir com as exigências de autonomia do analista, por exemplo, para definir e analisar uma nova estrutura de dados (Inmon 1996, Jarke et al. 1999, Quix 1999, Vassiliadis 2000a); (2) a abordagem que não oferece um componente que permita a partilha de metadados e opta pela autonomia dos componentes e utilizadores na interpretação e gestão de metadados locais, tornando frágil a metodologia *bottom-up* (Kimball et al. 1998); (3) a abordagem que defende um sistema distribuído que distingue, em cada componente, os metadados autónomos e os metadados partilhados que podem ser replicados para serem usados por todos os componentes do sistema, de modo a garantir as necessidades de autonomia na utilização de metadados e as necessidades de partilha de metadados ao longo de todo o sistema (Inmon et al. 2001).

Todavia, o debate prossegue e a investigação e a literatura procuram uma solução para gerir uma multiplicidade de metadados com inúmeras configurações e necessários ao desenvolvimento e evolução de sistemas com arquitecturas cada vez mais distribuídas e com uma vocação para a partilha de dados à escala global Vaduva e Dittrich 2001).

3 PROCESSO DE REFRESCAMENTO DOS RDW

O refrescamento dos RDW é um processo complexo que lida com os problemas associados à aquisição de dados do ambiente operacional para o ambiente *warehouse*. Neste capítulo, focaliza-se no problema do refrescamento dos RDW para avaliar os constrangimentos impostos pelo ambiente operacional e pelo ambiente *warehouse* e o impacto dos requisitos dos utilizadores, para descrever as tarefas de refrescamento. A secção 3.1 trata os constrangimentos e os requisitos dos utilizadores que influenciam o processo de refrescamento e a secção 3.2 centra-se na classificação das propriedades das fontes de dados traduzidas na autonomia, distribuição e heterogeneidade. Na secção 3.3, descrevem-se as várias tarefas de refrescamento, os objectivos que cumprem e os problemas com que lidam, incluindo a extracção de dados das fontes (subsecção 3.3.1), a limpeza de dados de fontes singulares (subsecção 3.3.2), a reconciliação e integração de dados (subsecção 3.3.3), o cálculo de dados derivados (subsecção 3.3.4), a gestão do histórico (subsecção 3.3.5) e o carregamento dos RDW (subsecção 3.3.6).[41]

3.1 Constrangimentos e Requisitos dos Utilizadores

O refrescamento dos RDW condiciona a efectiva utilização dos dados publicados pelos SDW. De facto, a qualidade dos dados disponibilizados para a tomada de decisão numa organização depende da capacidade dos SDW propagarem, num tempo razoável, as alterações ocorridas nos dados das diferentes fontes para os RDW (Bouzeghoub et al. 1999: 6.2).

Tornam-se, por isso, fundamentais as questões associadas, quer à gestão do intervalo de tempo entre o momento em que ocorrem alterações nos dados das fontes e o momento em que essas alterações são carregadas nos RDW, quer à decisão entre o carregamento total dos dados das fontes, em cada período de refrescamento, ou o carregamento incremental que permite a aquisição apenas das alterações ocorridas nos dados das fontes, entre dois períodos de refrescamento.

[41] As tarefas de refrescamento dos RDW são tratadas mais do ponto de vista dos problemas com que lidam. As soluções propostas na literatura são analisadas no capítulo 4, remetendo-se para Vassiliadis et al. a avaliação recente dos esforços no domínio da indústria (2005a).

Por um lado, a frequência de refrescamento dos RDW pode ser periódica, imediatamente após a ocorrência de alterações nos dados das fontes, por solicitação e/ou híbrida. Depende dos requisitos das aplicações em termos das exigências dos utilizadores relativas à idade dos dados *warehouse*, da frequência com que ocorrem alterações nos dados das fontes e consequente volume de dados envolvidos e dos aspectos do sistema e organizacionais, como as características de distribuição ou centralização da arquitectura dos SDW e o acesso autorizado às fontes de dados para suportar as rotinas de extracção incremental (Vavouras 2002: 42-3).

Por outro lado, o refrescamento dos dados *warehouse* lida com dados alterados nas fontes em diferentes proporções e ritmos de inserções, actualizações e eliminações. A aquisição apenas das alterações relevantes ocorridas nos dados das fontes reduz o tempo e o volume de dados envolvidos no processo de refrescamento, mas depende da disponibilidade de acesso e das capacidades das fontes de dados permitirem a detecção dessas alterações e do volume de dados alterados, podendo entrar em conflito com a performance esperada para o processamento de consultas aos RDW e o níveis desejados de consistência das vistas em termos da sintaxe e da uniformidade dos valores (Engström et al. 2003: 71-6).

Num estudo levado a cabo junto de 13 organizações norte-americanas que possuem SDW, os autores chegaram à conclusão que a frequência e duração das tarefas de refrescamento dos RDW variam com o número de fontes de dados e volume de dados alterados envolvidos, dependendo dos requisitos dos utilizadores relativamente à idade dos dados *warehouse* relevantes, dos constrangimentos impostos pelo ambiente operacional e pelo ambiente *warehouse* e do controlo dos custos (Mannino e Walter 2004).[42]

Na amostra estudada, os constrangimentos de curto prazo dominantes dizem respeito às fontes de dados em termos das restrições na frequência e tempo de extracção dos dados alterados e aos RDW no que refere às restrições devidas à capacidade de armazenamento, disponibilidade e uso dos servidores de dados *warehouse*. A integração das fontes de dados, a consistência das vistas e as expectativas dos utilizadores relativas à idade dos dados *warehouse* foram referidas como factores de curto prazo com menor influência no refrescamento dos RDW.

[42] A idade dos dados é uma dimensão de qualidade que surge na literatura associado aos conceitos de *freshness*, *currency*, *staleness* e *timeliness* (ver Bouzeghoub e Peralta 2004).

No longo prazo, o controlo dos custos associados ao tempo de execução do processo e aos conflitos com as transacções nas fontes de dados e com as operações de arquivo é o factor de maior impacto nas políticas de refrescamento adoptadas pelas organizações estudadas. Os maiores investimentos incidem na gestão da complexidade do processo de refrescamento, incluindo hardware, software de ETC e optimização.

Para além desse ponto de vista da prática das organizações, alguns autores têm sistematizado os constrangimentos e os requisitos dos utilizadores que influenciam o refrescamento dos RDW. A questão central na definição de uma estratégia de refrescamento tem-se situado na busca do equilíbrio entre a qualidade de serviço esperada pelos utilizadores e os custos associados aos recursos utilizados no processo (Vassiliadis 2000a: 2.4).[43]

Para os autores associados ao problema da manutenção das vistas materializadas esse equilíbrio é conseguido através de algoritmos que garantem um determinado grau de consistência das vistas materializadas e de um conjunto de técnicas e componentes de arquitectura para optimizar a manutenção incremental das vistas materializadas (Zhuge et al. 1995, 1996, 1997 e 1998; Labio e Garcia-Molina 1999; Quass et al. 1996).

Prosseguindo a investigação no domínio do problema da manutenção das vistas materializadas, Engström propõe uma abordagem baseada em custos e orientada à qualidade de serviço esperado pelos utilizadores (2002). O autor parte do pressuposto que existe um conjunto finito de políticas de manutenção das vistas materializadas para qualquer cenário realístico, demonstrando que a opção é influenciada pelos requisitos de qualidade de serviço esperada pelos utilizadores e pelas características do ambiente operacional e do ambiente *warehouse*.

O objectivo de qualquer política de manutenção das vistas materializadas é minimizar os custos relativos aos recursos utilizados na sua manutenção, como as capacidades de processamento (CPU e leitura e escrita nos discos rígidos), de armazenamento e de comunicação. cumprindo com a qualidade de serviço esperada pelos utilizadores (*idem*, 45).

O autor apresenta quatro dimensões da qualidade de serviço que as políticas de manutenção das vistas materializadas devem ter em conta: (1) o tempo de resposta, definido como a diferença de tempo entre a solicitação de uma consulta e a devolução

[43] Estão disponíveis outras propostas de classificação dos requisitos de qualidade dos dados, por exemplo, de Herfert (2001) e de Herfert e Hermann (2002).

do resultado; (2) a consistência das vistas, face às alterações ocorridas nas relações de base nas fontes de dados; (3) a idade das vistas, medida através da diferença entre o seu momento corrente e o último momento em que reflectiu os dados das fontes que lhe servem de base, incluindo todas as alterações ocorridas; (4) a disponibilidade das vistas.

Outros factores a considerar em cenários específicos de manutenção das vistas materializadas dizem respeito a características associadas às fontes de dados, como os índices de acesso e os tamanhos de página, e a características do ambiente *warehouse*, designadamente o comportamento do processamento das consultas em termos da frequência e da distribuição e da definição das propriedades das vistas (*idem*, 56).

Por seu lado, a equipa de investigadores do projecto DWQ liderada por Bouzeghoub propõe uma estratégia de refrescamento dos RDW orientada à qualidade para satisfazer os requisitos dos utilizadores em termos da disponibilidade, acessibilidade e idade dos dados publicados pelos SDW (Bouzeghoub et al. 2003: 50).[44]

Para estes autores, a definição de uma estratégia eficiente deve entender o refrescamento como um processo que não se restringe ao problema da manutenção das vistas materializadas e que compreende um conjunto de tarefas independentes e que podem ser ordenadas de diferentes maneiras.

Identificam um conjunto de factores a ter em conta na definição da estratégia: (1) os requisitos de aplicação, como a idade dos dados *warehouse*, a granularidade dos dados providenciados pelas fontes ou computados nas vistas e o tempo de computação das consultas e das vistas; (2) os constrangimentos das fontes de dados, como a disponibilidade e a frequência de alterações ocorridas nos seus conteúdos; (3) os limites do sistema, como as capacidades de armazenamento e de processamento dos repositórios de dados envolvidos (*idem*, 50).

O planeamento da estratégia deve ser adequado a cenários específicos de refrescamento e as tarefas devem ser ordenadas de acordo com as semânticas que se pretende atribuir ao processo e com a qualidade que se quer atingir. As semânticas do refrescamento são definidas como o conjunto de todas as decisões de desenho que

[44] Os objectivos de qualidade definidos para o processo de refrescamento incluem-se no *framework* de desenho e desenvolvimento de SDW proposto pelo projecto DWQ (ver Jarke et al. 1999). Theodoratos e Bouzeghoub apresentam um arquitectura para avaliar o grau de satisfação face à qualidade dos dados (1999 e 2001).

contribuem para providenciar dados relevantes para os utilizadores e que garantem os requisitos de qualidade (*idem*, 72).

A Tabela 3.1 sintetiza as quatro dimensões de qualidade que os autores consideram caracterizar todos os objectos dos SDW e a partir das quais se podem definir os factores de qualidade que constituem os requisitos dos utilizadores que devem ser incluídos no desenho do processo de refrescamento.

Dimensões de Qualidade	**Objectos do SDW**	**Factores Primários de Qualidade**	**Factores Derivados de Qualidade**	**Escolhas de Desenho**
Coerência (*Coherence*) – descreve a coerência lógica dos dados em termos da consistência	Fontes ODS Vistas	Disponibilidade de acesso de cada fonte de dados Tempo de resposta esperado para o processamento de dada consulta	Frequência da extracção em cada fonte de dados Tempo de resposta estimado para a extracção, para a integração e para a propagação das alterações	Granularidade dos dados nos vários objectos Política extracção e limpeza de dados Política integração de dados
Plenitude (*Completeness*) – descreve a percentagem de dados do mundo real que entraram para as fontes de dados	Fontes ODS	Disponibilidade de acesso de cada fonte de dados Duração histórico para cada RDW	Frequência da extracção em cada fonte de dados	Política extracção de dados Política integração de dados
Precisão (*Accuracy*) – descreve com precisão o processo de entrada de dados nas fontes	Fontes ODS Vistas	Disponibilidade de acesso de cada fonte de dados Duração histórico para cada RDW	Frequência da extracção em cada fonte de dados	Granularidade dos dados nos vários objectos Intervalo de histórico Política extracção de dados Política integração de dados
Idade dos dados (*Freshness*) – descreve o intervalo de tempo entre a entrada de dados nas fontes e o tempo de resposta a solicitações dos utilizadores	Fontes ODS Vistas	Disponibilidade de acesso de cada fonte de dados Idade dos dados esperada para o processamento de dada consulta Tempo de resposta estimado para a extracção, para a integração e para a propagação das alterações	Frequência de extracção em cada fonte de dados Real idade dos dados para o processamento de dada consulta Real tempo de resposta para o processamento de dada consulta	Política extracção de dados Política integração de dados Política refrescamento de dados

Tabela 3.1 Ligação entre factores de qualidade e escolhas de desenho

3.2 Classificação das Propriedades das Fontes de Dados

Usualmente, os SDW integram fontes de dados autónomas, distribuídas e heterogéneas. As fontes de dados podem ser entendidas como colecções de dados armazenados ou computados que se definem com base na combinação entre um componente de software e os dados, variando as tecnologias para aceder, descrever e armazenar os seus conteúdos (Katchaounov 2003: 7).

O processo de refrescamento interage com múltiplas fontes de dados, umas internas que documentam as operações diárias da organização e são geridas de forma independente pelas várias unidades da organização, como a produção, a gestão de *stocks*, a área comercial e logística, outras disponibilizadas por terceiros, como empresas que fornecem informação estatística e demográfica, listas de clientes alvo ou de diferentes segmentos de mercado alvo, e entidades reguladoras da indústria (Kimball et al. 1998: 336).

Podem variar desde simples ficheiros tabulares[45], acessíveis através de uma *Application Provider Interface* (API) do sistema de ficheiros, páginas de Internet, acessíveis através de um protocolo de transporte de um servidor de Internet, ficheiros de registo das acções - ficheiro *log* - de um servidor de Internet, acessível através de uma API proprietária, até bases de dados complexas, geridas por SGDB e acessíveis através de API standard como o *Call Level Interface* (X/Open 1995) ou API proprietárias como a interface de dois níveis da Oracle que inclui o *Oracle Common Interface* e o SQL* Net (Katchaounov 2003: 7).

O ambiente operacional dos SDW compreende, pois, fontes de dados autónomas, distribuídas e heterogéneas, propriedades que impõem constrangimentos a ter em conta na definição da estratégia de refrescamento (Bouzeghoub et al. 1997: 10). Por isso, tornam-se importantes as actividades de identificação e análise das fontes de dados para descrever as diferentes estruturas e semânticas dos seus dados (Kimball e Caserta 2004: 63-72).

Nas organizações, é comum as fontes de dados preexistirem ao desenvolvimento dos SDW. Por razões operacionais e organizacionais, as fontes de dados são normalmente geridas de forma autónoma por diferentes indivíduos e/ou unidades da

[45] Na literatura anglo-saxónica, o ficheiro tabular é usualmente designado por *flat file*. Trata-se de um ficheiro onde os dados são organizados e armazenados por coluna e por linha, para emular uma tabela de uma base de dados relacional. As colunas de um ficheiro tabular podem ser determinadas por comprimento fixo ou recorrendo a um caracter delimitador (Kimball e Caserta 2004 :26 e ss. e 91-3).

organização, cujas decisões sobre a escolha dos sistemas e modelos para armazenar os dados e dos métodos de manipulação e acesso aos dados variam entre si.

Estas decisões autónomas resultam, por exemplo, nas situações em que diferentes nomes e diferentes estruturas de dados nas várias fontes dizem respeito às mesmas entidades ou entidades semelhantes no mundo real, com base nas preferências de especificações e de terminologia dos utilizadores das aplicações do ambiente operacional. Ou nas situações de grande variação na frequência de alterações ocorridas nos dados das fontes e de disponibilidade de acesso aos seus dados (Winter e Meyer 2001: 24).

O nível de complexidade na interoperabilidade das fontes de dados no ambiente *warehouse* aumenta significativamente quando o número de fontes de dados para uso autónomo cresce em função do dinamismo das organizações, ao mesmo tempo que as contínuas alterações nos dados de fontes geridas de forma autónoma aumenta a complexidade das tarefas de refrescamento, entrando em conflito com as exigências de dados actualizados em tempo real para suportar tomadas de decisão rápidas face a modificações nas condições do mercado global (Bruckner et al. 2002: 317-8).

Várias propostas de classificação da autonomia das fontes de dados estão disponíveis na literatura, sobretudo no domínio da integração de fontes de dados. A proposta de Sheth e Larson (1990) é a mais completa e permite entender as decisões autónomas relativamente ao desenho, à comunicação, à execução e à disponibilidade da fontes de dados face a outro sistema, impondo constrangimentos quando participam no processo de refrescamento dos RDW (ver Tabela 3.2).

Desenho	Comunicação	Execução	Disponibilidade
Domínio do discurso (concepções do mundo real) Representação (modelo de dados, linguagem, ontologia) Interpretação semântica dos dados Restrições lógicas na gestão dos dados Associação e partilha com outros sistemas Funcionalidade do sistema Implementação (estruturas físicas de armazenamento, algoritmos)	Quando comunicar com outros componentes Como comunicar com outros componentes	Momento e ordem das operações que acedem aos dados Abortar as operações que não cumprem as restrições lógicas	Como partilha as operações que suporta e os dados que gere Quando e quanto partilha

Tabela 3.2 Autonomia das fontes de dados

Por outro lado, é também comum nas organizações os dados estarem distribuídos por múltiplas fontes, o que faz com que, por exemplo, possa existir demora

na comunicação com as fontes de dados ou se mantenham múltiplas cópias de dados suportados por diferentes estruturas e semânticas nas fontes e que têm que ser ajustadas aos requisitos definidos para os dados *warehouse* (Özsu e Valduriez 1999: 82-100).

As fontes de dados podem residir num único sistema de computadores ou estar distribuídas por múltiplos sistemas de computadores que coexistem num mesmo lugar ou se dispersam por múltiplos lugares, mas para os quais existe um meio físico de comunicação (rede de computadores).

Grande parte da heterogeneidade das fontes de dados é devida a diferenças tecnológicas, ou seja de hardware, dos sistemas de software e dos sistemas de comunicação, colocando problemas nos níveis de interoperabilidade do sistema, da sintaxe, da estrutura e da semântica (Sheth 1999: 8-9). Por exemplo, as fontes de dados podem representar de forma diferente as mesmas entidades do mundo real, podem usar diferentes métodos de acesso aos dados e diferentes sistemas operativos, podem possuir diferentes capacidades para detectar as alterações ocorridas nos seus conteúdos.

No domínio dos sistemas de informação, estão disponíveis várias classificações da heterogeneidade das fontes de dados. A Tabela 3.3 distingue três níveis de heterogeneidade, retirando contributos de vários autores, com destaque para as propostas de Sheth e Larson (1990), Ouksel e Sheth (1999), Sheth (1999) e Hakimpour e Geppert (2001).

Plataforma	Sistema	Dados
Sistemas operativos Hardware Sistemas de comunicação e protocolos de rede Métodos que providenciam o acesso programático aos dados Representação física dos dados	Modelos de dados (Conceitos para modelar as entidades do mundo real; Restrições de integridade) Linguagens de acesso aos dados (Capacidades das fontes de dados)	Esquema (Nome e estrutura) Construções de modelação (Por exemplo, uma concepção modelada como um atributo num esquema e como uma tabela noutro esquema) Semântica (Por exemplo, o mesmo nome para coisas diferentes ou diferentes nomes para a mesma coisa) Sintaxe (Formato e precisão)

Tabela 3.3 Heterogeneidade das fontes de dados

Um dos problemas críticos do processo de refrescamento consiste na detecção e isolamento das alterações ocorridas nos dados das fontes logo na tarefa de extracção, de modo a reduzir o volume de dados a migrar para os RDW. A operação depende das disponibilidades de acesso e das capacidades das fontes de dados.

A heterogeneidade das capacidades das fontes de dados tem sido uma das questões que mais tem preocupado os autores. Existem algumas propostas de classificação, optando-se por apresentar a proposta de Bouzeghoub et al. que sintetiza

propostas anteriores (2003: 57 e ss).[46] Distinguem-se as fontes de dados cooperativas que providenciam um ou mais mecanismos que permitem a detecção automática de alterações nos seus conteúdos e a respectiva notificação, como os *triggers* e as regras *Event-Condition-Action* (ECA), e as fontes de dados não cooperativas que não oferecem um mecanismo para detectar e notificar automaticamente as alterações ocorridas nos seus conteúdos (ver Tabela 3.4).

<table>
<tr><th colspan="2">Fontes Cooperativas</th><th>Fontes Não Cooperativas</th></tr>
<tr><td colspan="2">Fontes Replicadas
Suportam serviço de replicação que pode ser utilizado para analisar e detectar as alterações de interesse para os utilizadores</td><td rowspan="2">Fontes Snapshot
Não têm capacidade selectiva. Apenas permitem a descarga completa do seu conteúdo (bulk)
Fontes Logged
Mantêm as suas acções num ficheiro log de onde se podem extrair ou consultar as alterações e operações relevantes
Fontes Consultáveis (queryable)
Oferecem um interface de processamento de consultas aos dados que tornam possível a detecção de alterações relevantes, recolhendo os dados periodicamente e comparando-os com a última versão
Fontes Específicas
Apresentam capacidades específicas que podem ser usadas na detecção de alterações (por exemplo, escrever a descrição das suas acções como suporte a actividades de auditoria que podem ser usadas para detectar as alterações relevantes, controlar a ocorrência de alterações com elementos internos temporais em todos os registos (timestamps) e ser capaz de isolar e providenciar os registos alterados num único passo)</td></tr>
<tr><td>Fontes Activas
Providenciam triggers e outras capacidades activas que podem detectar automaticamente alterações de interesse e notificar de imediato outros componentes</td><td>Fontes de acção externa (callback)
Acções de notificação podem ter efeitos sobre sistemas externos à fonte de dados onde ocorreram as alterações

Fontes de acção interna
Acções geradas pela capacidade activa não têm um efeito sobre um sistema de bases de dados externo</td></tr>
</table>

Tabela 3.4 Classificação das capacidades das fontes de dados

3.3 As Tarefas do Processo de Refrescamento

O processo de refrescamento dos RDW pode ser decomposto num conjunto de tarefas independentes e organizadas para cenários específicos de refrescamento: (1) a **extracção** dos dados das fontes, detectando as alterações ocorridas; (2) a **limpeza** das anomalias nos dados, removendo os erros e as inconsistências; (3) a **reconciliação** das diferenças de estrutura e de semântica e a **integração** dos dados num formato comum e consolidado; (4) a **gestão do histórico**, mantendo a perspectiva temporal dos dados alterados; (5) o **cálculo dos dados derivados,** enriquecendo os dados operacionais com

[46] Ver outras classificações em Zhou et al. (1995), Widom (1995), Koschel e Lockemann (1998) e Engström (2002).

a apresentação de formatos analíticos de dados; (6) o **carregamento** dos RDW, computando e propagando as alterações ao longo da hierarquia das camadas de dados e propagando os dados sumariados para os *data marts*.[47]

As tarefas são executadas de acordo com uma ordem e um calendário definidos a partir da análise dos constrangimentos das fontes, dos requisitos dos utilizadores e das definições das vistas, articulando as técnicas e os métodos que podem ser utilizados, como o paralelismo, a sincronização, a manutenção de informação de controlo de execução, o tratamento de excepções e a recuperação de situações de falha.

As tarefas de refrescamento podem ser executadas de forma completamente assíncrona, com um alto nível de paralelismo na execução das rotinas que asseguram a preparação dos dados das diferentes fontes, atendendo à disponibilidade específica de cada fonte de dados e à técnica de extracção utilizada. Dependendo do contexto de cada SDW, algumas tarefas podem ainda ser agrupadas e combinadas numa só, enquanto que outras podem ser decompostas em várias sub-tarefas de execução independente (Bouzeghoub et al 1999).

Por outro lado, as tarefas de preparação dos dados extraídos das fontes são, normalmente, cumpridas numa área intermédia de processamento temporário de dados (Kimball et al. 1998). Persistir os resultados de cada uma das tarefas num meio físico de armazenamento temporário, como por exemplo ficheiros tabulares do sistema operativo ou tabelas num SGBD relacional, pode ser útil: (1) na recuperação de falhas, de modo a evitar a repetição das tarefas anteriormente executadas; (2) na sincronização das tarefas executadas individualmente sobre os dados de cada uma das fontes; (3) na organização dos dados de um modo que permita optimizar a execução das rotinas de refrescamento.

3.3.1 Extracção de Dados

A extracção é a primeira tarefa do processo de refrescamento e consiste em extrair os dados das fontes para posteriores tarefas de preparação e propagação para o ambiente *warehouse*. Exige a identificação e documentação das fontes de dados candidatas, a análise da estrutura e dos conteúdos das fontes de dados e a identificação dos requisitos dos utilizadores em termos das necessidades de informação (ver Kimball e Caserta 2004: 4, 63-73).

[47] Seguem-se de perto as pistas deixadas por Bouzeghoug et al. (1999) sobre o entendimento do refrescamento como um *workflow*, não deixando de se recorrer a outros autores sempre que se justificar.

A tarefa lida com os problemas associados à detecção, computação, extracção e escrita das alterações relevantes ocorridas nos dados das fontes, dependendo das propriedades das fontes de dados em termos das linguagens de acesso, das capacidades que podem ser exploradas para a detecção das alterações ocorridas nos seus conteúdos e do momento e modo como se deve processar a comunicação.

Existem dois tipos de técnicas para extrair os dados das fontes: (1) as técnicas de extracção total em que se extraem todos os dados das fontes, sem que sejam detectadas as alterações ocorridas desde o último período de extracção; (2) as técnicas de extracção incremental em que se detectam e extraem apenas as alterações ocorridas nos dados das fontes, desde o último período de extracção (Vavouras 2002: 38-41).[48]

Um aspecto crítico nas técnicas incrementais é utilizar os métodos que permitem computar as alterações relevantes ocorridas nos dados das fontes tão cedo quanto possível (ver Ram e Do 2000, Rosana et al. 2003). Tem vantagens em termos de tempo, custos e volume de dados a ser migrado para a área intermédia de processamento temporário de dados, mas está dependente da disponibilidade de acesso e das capacidades das fontes de dados, do volume de dados a extrair das fontes de dados, da frequência com que ocorrem alterações nos dados das fontes e dos requisitos dos utilizadores em termos da idade e do histórico dos dados *warehouse*.

A computação das alterações relevantes pode ser executada através de métodos externos ou de métodos intrusivos (Fabret et al. 1997: 14-7).

Os métodos externos executam a computação das alterações sem usar qualquer capacidade das fontes de dados e socorre-se da técnica do *snapshot*. A estratégia consiste em calcular o diferencial de alterações com base num algoritmo, comparando dois estados sucessivos das fontes de dados (*snapshots*).[49] Estes métodos reduzem a sobrecarga na fonte de dados, mas envolvem a transmissão de um grande volume de dados, maiores custos e perda parcial do histórico.

Por seu lado, os métodos intrusivos exploram as capacidades das fontes de dados para computar as alterações relevantes, podendo recorrer-se a diferentes soluções, como por exemplo: (1) os *triggers* definidos no esquema das fontes de dados, que detectam as operações que alteram os dados (inserção, actualização e eliminação) e descrevem o

[48] Cabe a um componente especializado implementado nos *wrappers* detectar, extrair e propagar as alterações relevantes ocorridas nos dados das fontes, para além da função tradicional dos *wrappers* de aceder às fontes de dados e apresentar os dados extraídos num formato comum (Bouzeghoug et al. 2003: 54).

[49] Ver exemplos destes algoritmos em Labio e Garcia-Molina (1996).

diferencial das alterações numa tabela auxiliar, reduzindo-se o volume de dados transmitido, mas tornando as aplicações operacionais mais lentas e as fontes de dados podem recusar a intrusão ao nível do esquema; (2) os mecanismos de replicação, que propagam as alterações periodicamente, por solicitação ou com base em critérios específicos (por exemplo, quando ocorre um determinado número de alterações), tornando-se a solução menos intrusiva; (3) as tabelas de dados operacionais que possuem colunas com selos temporais (*timestamps*) que especificam o tempo e a data da última alteração numa dada linha, permitindo isolar e fornecer os registos alterados num único passo; (4) os ficheiros que mantêm o registo das acções das aplicações operacionais (*log*) que contêm a sequência das alterações ocorridas nos dados das fontes, mas que são de acesso restrito e de difícil interpretação.

A decisão por uma determinada técnica de extracção de dados pode também ser condicionada pelo volume de dados a extrair de cada uma das fontes e a frequência com que ocorrem alterações nos seus conteúdos (proporções e ritmos de inserções, actualizações e eliminações), podendo optar-se por extrair os objectos de um dado tipo de modo incremental e os objectos de um outro tipo de modo total.

Também os requisitos dos utilizadores em termos da idade dos dados *warehouse* condicionam a decisão sobre a frequência com que se extraem os dados das fontes e sobre as técnicas de extracção a utilizar. Quando os requisitos de idade dos dados são muito elevados (por exemplo, 1 minuto de idade), devem ser detectadas e extraídas as alterações que ocorrem nos dados das fontes tão cedo quanto possível, recorrendo-se, por exemplo, às capacidades activas das fontes de dados (ver Bruckner et al. 2002).

Do mesmo modo, os requisitos dos utilizadores no que se refere ao histórico dos dados *warehouse* podem favorecer a decisão por uma determinada técnica de extracção. Quando os requisitos de histórico dos dados são elevados (por exemplo, preservar o histórico de todas as alterações), é necessário recorrer a técnicas de extracção incremental que permitam detectar todas as alterações que ocorrem nos dados das fontes para preservar o histórico completo desses dados. Pelo contrário, a utilização de uma técnica de extracção total conduz a uma perda parcial do histórico dos dados, permitindo apenas preservar sequências discretas dos estados das fontes de dados (estados observáveis) e perdendo-se os estados intermédios (Vavouras 2002: 41).

Por outro lado, as propriedades de autonomia das fontes de dados podem impor limitações nas decisões sobre a técnica e método de extracção a utilizar, sobre as capacidades das fontes de dados a explorar e sobre o período e frequência com que pode

ocorrer a extracção. Podem, por exemplo, limitar a intrusão e a sobrecarga provocada nas fontes de dados e exigir que a extracção dos dados de uma fonte se processe diariamente, no período de menor actividade das fontes de dados e que a extracção dos dados de outra fonte se processe semanalmente, no final de cada semana de actividade.

A autonomia de cada fonte de dados no que se refere ao período e frequência com que pode ocorrer a extracção torna necessária a sincronização durante a reconciliação e integração de dados, por forma a que os dados propagados para os RDW possam reflectir um estado globalmente consistente e completo (Bouzeghoub et al. 2003: 50).

3.3.2 Limpeza de Dados

A limpeza de dados tem por objectivo detectar e remover os erros e as inconsistências dos dados extraídos das fontes, de modo a melhorar a sua qualidade e permitir a integração nos RDW.[50] A tarefa lida com os problemas dos dados *sujos*, ou seja, com as anomalias que se verificam nos dados extraídos das fontes de dados.

A literatura mais recente sobre a limpeza de dados tem adoptado a classificação proposta por Rahm e Do para distinguir os problemas dos dados *sujos* de uma fonte singular e os problemas dos dados *sujos* de múltiplas fontes, nos níveis das instâncias e dos esquemas (2000).[51]

De acordo com o entendimento do processo de refrescamento como um *workflow*, as rotinas de limpeza podem ocorrer em diferentes momentos do processo de refrescamento, distinguindo-se: (1) as rotinas de limpeza que ocorrem imediatamente após a extracção de dados, para detectar e remover as anomalias específicas de cada uma das fontes de dados; (2) as rotinas de limpeza que ocorrem na integração dos dados, para detectar e remover as anomalias que resultam da captura de dados de duas ou mais fontes com propriedades diferentes (Bouzeghoub et al. 1999 e Vavouras 2002: 33). Por isso, opta-se por tratar aqui o primeiro conjunto de problemas dos dados *sujos*, não deixando de dar conta de aspectos globais associados à estratégia da limpeza de dados, mas remetendo-se para a subsecção que se segue o segundo conjunto de problemas dos dados *sujos*.

[50] Na literatura, esta tarefa é habitualmente referida por *data cleansing*, *data cleaning* ou *scrubbing*.

[51] Ver as ferramentas de limpeza de dados propostas por Galhardas et al. (2000a e 2000b), Lee et al. (2000), Raman e Hellerstein (2001).

Os problemas dos dados *sujos* de uma fonte singular incluem as anomalias no nível das instâncias e no nível do esquema.

No nível das instâncias, distinguem-se: (1) no atributo (campo), os valores em falta, os erros de fonética, os valores abreviados, os valores embebidos noutros valores ou inseridos em campos errados; (2) no registo, as dependências entre atributos violadas; (3) no tipo de registo, as transposições de palavras, os registos duplicados e os registos contraditórios; (4) na fonte, referências incorrectas entre registos de diferentes tipos. No nível do esquema, distinguem-se: (1) no atributo (campo), os valores ilegais; (2) no registo, as dependências entre atributos violadas; (3) no tipo de registo, a violação da unicidade; (4) na fonte, a violação das restrições de integridade.

As anomalias nos dados de uma fonte singular dizem respeito aos erros e inconsistências que não podem ser prevenidos no nível do esquema. Resultam das limitações do modelo de dados, da não estruturação do esquema ou das reduzidas restrições de integridade definidas para controlar a introdução de valores permitidos.

Nas fontes menos estruturadas, como os ficheiros tabulares, as reduzidas restrições relativas à entrada de dados aumentam a probabilidade de erros e inconsistências e, muitas vezes, mesmo nas fontes de dados geridas por SGBD, a sobrecarga adicional provocada pela certificação de conformidade dos dados exige um compromisso entre a eficiência de resposta da fonte de dados e a qualidade dos dados conseguida.

Os métodos mais comuns utilizados na limpeza de dados de fontes singulares compreendem os métodos estatísticos que permitem detectar e remover os valores inválidos e completar os valores em falta, a análise gramatical que permite detectar e corrigir erros de sintaxe, a conversão dos diferentes formatos de dados no formato comum definido para os dados *warehouse*, e a procura de termos válidos em dicionários de dados para corrigir os erros de sintaxe e validar informação específica de um dado domínio de aplicação (Müller e Freytag 2003: 13-5, Bouzeghoub et al. 2003: 63-4). Acresce ainda a imposição de restrições de integridade para detectar e remover as violações das regras de um dado domínio de aplicação (Embury et al. 2001, Kimball e Caserta 2004: 135-6).

As decisões sobre a estratégia de limpeza colocam questões associadas, por um lado, à definição de critérios de limpeza que permitam melhorar a qualidade dos dados e, por outro, à eficiência das rotinas que aplicam esses critérios em grandes volumes de dados (Müller e Freytag 2003: 3 e 10-2, Kimball e Caserta 2004 18-9).

De facto, os critérios de limpeza de dados são heurísticas normalmente complexas e muito específicas de um dado domínio de aplicação. Definir os critérios de limpeza adequados é uma tarefa difícil que exige o conhecimento profundo do domínio de aplicação em causa para identificar os tipos de erros e inconsistências a serem removidos. Depois de traduzir o conhecimento do domínio de aplicação em critérios de limpeza apropriados é necessário, muitas vezes, avaliar a sua eficácia quando aplicados sobre os dados e refiná-los iterativamente, de acordo com o nível desejado de qualidade de dados (Rahm e Do 2000: 5-8, Müller e Freytag 2003: 10-2).[52]

Torna-se, pois fundamental a definição de uma estratégia eficaz de limpeza de dados que compreenda os seguintes principais passos: (1) a análise de dados focalizada quer no perfil de cada atributo (*data profiling*), quer nas relações entre os atributos (*data mining*), para identificar os tipos de erros e inconsistências a serem removidos; (2) a definição da sequência das operações de limpeza, enquadrada no entendimento do refrescamento como um *workflow*, e dos métodos para detectar e remover os erros e inconsistências identificadas, começando-se pelas rotinas de limpeza de dados das fontes singulares e prosseguindo-se com as transformações e os mapeamentos, tendo em conta o número de fontes de dados envolvidas e seu grau de heterogeneidade e a *sujidade* dos dados; (3) a verificação da precisão e eficácia da estratégia de limpeza definida, através da avaliação numa amostra; (4) a execução das rotinas de limpeza de dados; (5) a substituição dos dados sujos de cada fonte pelos dados limpos, para melhorar a qualidade dos dados nas fontes e evitar a repetição das operações em futuros refrescamentos (Rahm e Do 2001: 5-6, Han e Kamber 2001: 109-12, Müller e Freytag 2003: 10-2).

Por outro lado, os cada vez mais curtos intervalos de refrescamento e o volume de dados envolvido fazem da limpeza de dados um tarefa tecnicamente difícil, jogando-se os requisitos dos utilizadores em termos da qualidade dos dados e o tempo e recursos computacionais consumidos na detecção e remoção das anomalias (Kimball e Caserta 2004: 120-1).

A comparação de objectos é uma das operações mais dispendiosas das rotinas de limpeza de dados, consumindo a maior parte do tempo de execução. Reduzir o tempo

[52] Por vezes, os critérios de limpeza adoptados não conseguem cobrir eficazmente a totalidade dos dados extraídos de uma fonte singular, existindo anomalias que não podem ser detectadas e removidas automaticamente pelas rotinas de limpeza. A correcção destas anomalias exige, pois, a intervenção de um especialista do domínio da aplicação (Galhardas 2001: 13-4 e 35, Müller e Freytag 2003: 3 e 12).

despendido na comparação de objectos é uma questão crítica na redução do tempo total de execução das rotinas de limpeza (Sung et al. 2002: 77).[53] A estratégia deve assentar num compromisso entre a qualidade de dados a atingir e o tempo e recursos computacionais envolvidos na execução das rotinas de limpeza de dados.

3.3.3 Reconciliação e Integração de dados

Depois de se extrair e limpar os dados de cada uma das fontes, segue-se a reconciliação e integração dos dados no formato uniforme e consolidado. A tarefa de reconciliação e integração lida com os dados provenientes de múltiplas fontes de dados, colocando-se os problemas associados à heterogeneidade dos modelos de dados, do desenho dos esquemas e dos dados.

De facto, a heterogeneidade das múltiplas fontes de dados decorrente da sua gestão autónoma agrava as anomalias que se verificam individualmente em cada uma das fontes de dados, acrescentando outras anomalias que ocorrem, quer no nível das instâncias, quer no nível dos esquemas (Rahm e Do 2001: 4-5).

No nível das instâncias, distinguem-se: (1) as diferentes representações de valores nas diferentes fontes de dados; (2) as diferentes interpretações de valores nas diferentes fontes de dados; (3) os diferentes níveis de detalhe dos dados nas diferentes fontes de dados e (4) os diferentes momentos a que os dados se referem nas diferentes fontes de dados. No nível dos esquemas, distinguem-se: (1) o mesmo nome usado para diferentes coisas (homónimos) ou diferentes nomes usados para a mesma coisa (sinónimos); (2) as diferentes construções de modelação para modelar as mesmas entidades do mundo real nas diferentes fontes de dados (conflitos estruturais).[54]

A resolução desses problemas requer a integração dos dados e a limpeza das anomalias traduzida na reconciliação dos conflitos nos níveis dos esquemas e das instâncias. Implica um conjunto de transformações de dados definidas numa linguagem apropriada para permitir os mapeamentos dos dados das múltiplas fontes para o formato uniforme e consolidado definido para os RDW (Bouzeghoub et al. 1999: 6).[55]

---

[53] A comparação é uma das operações mais comuns da limpeza de dados. É utilizada, por exemplo, na procura de termos inválidos (comparando-os com termos aproximados de um dicionário de dados), de valores sinónimos (comparando-os com termos de um dicionário de dados) e na identificação de objectos duplicados (comparando os objectos de um dado tipo).

[54] Ver a classificação dos conflitos estruturais que decorrem da integração de esquemas de Batini et al. (1986), Spaccapietra et al. (1992).

[55] Sobre as abordagens de integração de esquemas e de dados, ver a secção 2.2.

Durante a integração, é necessário juntar os objectos do mesmo tipo por forma a que possam ser propagados em conjunto para as mesmas estruturas de dados dos RDW. Torna-se, por isso, fundamental identificar os objectos duplicados, contribuindo para a melhoria da qualidade dos dados. A existência de objectos duplicados cujos valores não são exactamente os mesmos (duplicados inexactos) fazem desta operação um problema complexo (Cali et al. 2003: 378).[56]

Estão disponíveis na literatura as técnicas e métodos que procuram identificar os objectos duplicados através de uma dada função ou critério de similaridade, destacando-se os métodos de vizinhança ordenada e de vizinhança com múltiplas passagens e as técnicas de segmentação (ver Hernández e Stolfo 1995 e Monge e Elkan 1997), e distinguindo-se as abordagens que são dependentes do domínio de aplicação (por exemplo, Low et al. 2001) e as abordagens independentes do domínio de aplicação (por exemplo, Ananthakrishna et al. 2002).

Por outro lado, a autonomia das fontes de dados coloca o problema das alterações nos dados de cada uma das fontes poderem ter sido detectadas e extraídas em diferentes períodos e com diferentes frequências, tornando complexa a identificação de objectos duplicados e exigindo a sincronização dos conteúdos extraídos de cada uma das fontes de dados, por forma a que os dados propagados para os RDW possam reflectir um estado globalmente consistente e completo (Bouzeghoub et al. 2003: 67).

Acresce ainda o facto de as alterações que ocorrem nos dados das fontes serem integradas com os dados anteriormente carregados nos RDW. Torna-se necessário garantir o mapeamento dos objectos de cada uma das fontes de dados para os objectos anteriormente carregados nos RDW, após se aplicarem os métodos apropriados de detecção de objectos duplicados. Nas fontes de dados que mantêm chaves locais dos seus objectos pode-se, por exemplo, adicionar um identificador da fonte à chave local de cada objecto ou atribuir uma nova chave única a cada objecto (Vavouras 2002: 35-6).

3.3.4 Cálculo de Dados Derivados

O cálculo de dados derivados é a tarefa de refrescamento responsável por enriquecer os dados operacionais, de modo a melhorar a análise nos processos de decisão. Nesta tarefa, coloca-se o desafio de lidar com camadas de dados dependentes

[56] Trata-se de um problema usualmente referido como *inexact duplicates problem*, *fuzzy duplicates problem*, *object matching problem*, *object identity problem*, *instance identification problem*, *duplicate elimination problem*, *record linkage problem* ou *merge/purge problem*.

entre si, isto é, com camadas de dados cujos conteúdos são calculados a partir dos conteúdos de uma ou mais camadas de dados que lhes servem de base.[57]

Enquanto que, nos sistemas operacionais, os dados derivados são usualmente calculados no momento em que são solicitados, no ambiente *warehouse*, é usual calcularem-se e armazenarem-se, antecipadamente, os dados derivados para melhorar a performance do processamento de consultas. Por isso, quando os conteúdos de uma camada de base são alterados, é necessário computar as alterações que devem ser aplicadas nos conteúdos das respectivas camadas de dados derivados (Vavouras 2002: 34).

Um dos cálculos de dados derivados mais usuais no processo de refrescamento é o cálculo de dados sumariados. Os dados *warehouse* podem ser organizados em diferentes camadas e de acordo com o nível de detalhe, desde as camadas de base que contém os dados no mais alto nível de detalhe até às camadas especializadas que contêm dados muito sumariados (Inmon 1996: 37-38, Jarke e Quix 2003: 10-2).

Para melhorar as actividades de análise dos dados publicados pelos SDW, podem ser calculados outros dados derivados, incluindo: (1) os cálculos aritméticos e baseados em regras, como, por exemplo, a idade dos indivíduos com base na diferença entre a data corrente e a data de nascimento; (2) completar os dados das fontes internas com os dados das fontes externas para calcular, por exemplo, o rendimento esperado das famílias, tendo em conta os dados estatísticos providenciados por uma entidade competente; (3) completar os dados extraídos das fontes com informação temporal, por exemplo a data em que os dados foram alterados pela última vez; (4) completar os dados operacionais com informação associada às fontes de dados, como adicionar um nome para designar a fonte de onde provêm os dados (Vavouras 2002: 33-4).

3.3.5 Gestão de Histórico

A gestão de histórico é a tarefa responsável por preservar o histórico dos dados, isto é, garantir que os dados *warehouse* traduzem a evolução dos dados das fontes, permitindo a análise de tendências. Lida com o problema das fontes de dados armazenarem, usualmente, apenas dados correntes recentes.

Nos SDW, pode ser necessário gerir o histórico dos dados *warehouse*, em duas situações: (1) quando os dados das fontes são alterados e se torna importante manter os

[57] Esta tarefa de refrescamento é por vezes referida por *completion* (Vavouras 2002: 33-4).

valores antes e após essas alterações; (2) mesmo quando não ocorrem alterações no dados das fontes, mas existem dados *warehouse* definidos com base numa referência temporal e que variam à medida que o tempo avança, como é o caso da definição da idade de um indivíduo a partir da respectiva data de nascimento (Yang 2001: 2).[58]

A gestão do histórico nos SDW pode variar entre dois extremos: (1) o armazenamento de simples cópias das fontes de dados, sem que os RDW providenciem informação temporal das alterações que ocorrem nos dados das fontes; (2) a manutenção do histórico de todas as alterações ocorridas nos dados das fontes. As opções estão associadas aos requisitos definidos para cada uma das aplicações do ambiente *warehouse*, podendo preservar-se o histórico completo dos objectos de um dado tipo e manter apenas os valores correntes dos objectos de outro tipo (Vavouras 2002: 36-7).[59]

Por outro lado, as fontes de dados podem possuir capacidades para detectar todas as alterações que ocorrem nos seus conteúdos, tornando possível a preservação completa do histórico ou, pelo contrário, podem exigir a utilização de um método externo que apenas permite detectar as alterações ocorridas em sequências discretas dos estados das fontes de dados, conduzindo a uma perda parcial do histórico dos dados *warehouse* (Bouzeghoub et al. 1997: 7).

3.3.6 Carregamento dos RDW

O passo final do processo de refrescamento consiste no carregamento dos dados reconciliados e integrados nos diferentes RDW, de modo a reflectirem as alterações ocorridas nos dados das fontes. A tarefa lida com um grande volume de dados e exige um conjunto de procedimentos adicionais, podendo ser muito demorada e entrar em concorrência com o processamento de consultas aos dados *warehouse*, uma vez que os períodos em que os RDW não são consultados são, normalmente, muito curtos (Chandhury e Dayal 1997: 67).

A técnica mais usual para popular os RDW consiste na utilização de utilitários de carregamento massivo providenciados pelos SGBD que gerem o armazenamento e

[58] Yang e Widom propõem um *framework* de manutenção temporal das vistas materializadas para as relações base em fontes de dados temporais ou não temporais, adoptando a política de manutenção imediata para a actualização de dados temporais alterados e a política de manutenção diferida para a actualização de dados temporais que se alteram à medida que o tempo avança, admitindo a importância do conceito de manutenção autónoma das vistas materializadas temporais quando os dados base temporais não estão disponíveis nas fontes de dados não temporais (1998).

[59] Ver as técnicas de *Slowly Changing Dimensions* e as questões associadas à correcção de erros nos dados em Kimball e Caserta (2004: 183-95)

utilização dos dados *warehouse*.[60] Trata-se de uma técnica que permite que múltiplas inserções de dados sejam agrupadas num único lote, reduzindo-se os custos envolvidos na comunicação com os RDW e nas operações de *input*/*output* dos discos rígidos onde os dados são armazenados fisicamente (Cai et al. 2005: 3). Alguns dos procedimentos adicionais que ocorrem durante o carregamento, como a verificação de restrições de integridade e a reorganização dos índices, também podem ser agrupados e processados em lote (Wiener e Naughton 1994: 2, Amer-Yahia e Cluet 2004: 239-40).[61]

Esses utilitários têm que lidar com um grande volume de dados, tornando-se o carregamento sequencial muito longo. O paralelismo permite optimizar a performance de carregamento, designadamente o paralelismo nas operações de *input/output* aos discos, o paralelismo entre diferentes transacções e o paralelismo entre diferentes operações de uma mesma transacção (Silberschatz et al. 2006: 809-26).

Uma técnica usual de paralelismo consiste na utilização de particionamento de dados para possibilitar que as transacções e os procedimentos adicionais sejam aplicados, em paralelo, sobre as partições que precisam de ser actualizadas, reduzindo-se os custos das operações de *input/output* aos discos e o impacto provocado no processamento de consultas aos dados *warehouse* e sem afectar os dados e os índices de outras partições (Kimball e Caserta 2004: 291-2, Silberschatz et al. 2006: 810).

Alguns desses utilitários conseguem optimizar o carregamento prescindindo das operações de recuperação executadas nativamente pelo SGBD (Kimball e Caserta 2004: 226-7). Contudo, é conveniente recorrer às técnicas que permitem retomar o carregamento quando este é interrompido, evitando o esforço da sua reiniciação e resolvendo eventuais situações em que os RDW podem ficar num estado inconsistente ou incompleto, devido à interrupção (Labio et al. 2000: 46).

Por outro lado, os utilitários podem permitir fazer o carregamento total ou o carregamento incremental.[62]

O carregamento total tem a vantagem de poder ser tratado como um longa transacção em lote que, quando termina, produz um novo repositório, ao mesmo tempo que possibilita o processamento de consultas os dados do repositório corrente, durante a

[60] Esses utilitários são designados por *bulk/batch load utilities*.

[61] Nas situações de carregamento massivo de dados, é necessário optimizar a reorganização dos índices, recorrendo às técnicas de carregamento massivo de índices (ver, por exemplo, Fenk et al. 2000, Bercken e Seeger 2001).

[62] Ver os utilitários referidos por Vavouras (2002: 37-8) e Cai et al. (2005: 2).

transacção. Contudo, o carregamento total pode exigir um longo período de transacção, mesmo quando se utilizam as técnicas de paralelismo. O carregamento incremental, por sua vez, lida com transacções de carregamento mais pequenas para evitar entrar em conflito com o processamento das consultas aos dados *warehouse,* exigindo que a sequência das transacções seja coordenada, de modo a assegurar a consistência dos dados derivados e dos índices relativamente aos dados base (Chaudhuri e Dayal 1997: 67-8).

As rotinas de carregamento são aplicadas sobre diferentes camadas de dados armazenadas e geridas, eventualmente, por diferentes SGBD e, por isso, estão sujeitas a diferentes complexidades e requisitos e dependem do volume de dados envolvidos.

O carregamento dos ODS compreende um conjunto de transacções que actualizam frequentemente um pequeno volume de dados. Os requisitos dos utilizadores no que trata à idade dos dados faz aumentar a frequência dos períodos de refrescamento e, consequentemente, a probabilidade das transacções que actualizam os dados colidirem com as transacções que consultam esses mesmos dados, tornando complexa a coordenação destas transacções (Fabret et al. 1997: 6).

O carregamento do DW global compreende usualmente uma grande carga de transacções que actualizam, periodicamente, grandes volumes de dados. Os requisitos dos utilizadores no que se refere à disponibilidade dos dados e ao tempo de resposta no processamento das consultas impõem limitações no tempo e recursos computacionais consumidos, tornando-se necessário minimizar a interferência provocada nos contínuos acessos aos dados por parte dos utilizadores (Quass e Widom 1997: 1).

Por sua vez, o carregamento dos *data marts* envolve usualmente múltiplas transacções que actualizam pequenos volumes de dados derivados (Fabret et al. 1997: 6).

Os RDW podem conter múltiplas vistas materializadas que suportam o processamento de consultas mais frequentes aos dados derivados, permitindo, deste modo, optimizar o processamento de consultas (Engström 2002:21). As vistas materializadas podem ser mantidas através de técnicas de recomputação total dos dados base sobre as quais foram definidas ou através de técnicas de recomputação

incremental, ou seja, da recomputação das alterações a serem aplicadas às vistas materializadas.[63]

São muito comuns as situações em que se aplica a heurística da inércia, isto é, em que apenas é necessário actualizar uma parte das vistas materializadas, face às alterações que ocorrem nos dados de base, tornando-se mais eficiente a utilização das técnicas de manutenção incremental (Gupta et al. 1993: 193-4). Essas técnicas são igualmente desejáveis quando é necessário preservar o histórico, pois o volume total dos dados das vistas materializadas é cada vez maior, à medida que o tempo avança (Yang 2001: 22).[64]

[63] Sobre manutenção das vistas materializadas ver, por exemplo, Gupta e Munick (1995 e 1999), Gupta et al. (2001), Sartori (2001), Liu et al. (2002). Sobre a optimização da manutenção das vistas materializadas assente em técnicas de particionamento, ver Folkert et al. (2005). Os projectos clássicos de manutenção das vistas materializadas serão tratadas em 4.1.

[64] Nas situações em que não é necessário manter o histórico dos dados pode ser mais eficiente a recomputação total, por exemplo, quando os dados de base são totalmente eliminados das camadas inferiores e a recomputação total permite determinar rapidamente o estado "vazio" das vistas materializadas, evitando-se o esforço de computar as alterações ocorridas (ver Gupta et al. 1993: 193).

4 ABORDAGENS DE REFRESCAMENTO DOS RDW

Estão disponíveis na literatura diferentes abordagens do problema do refrescamento dos RDW. Neste capítulo, analisa-se a literatura mais relevante produzida no domínio específico do refrescamento dos RDW e os contributos do domínio do processo de ETC, discutindo-se as principais abordagens e as soluções propostas, numa perspectiva crítica e comparada. Distinguem-se as abordagens de manutenção das vistas materializadas (secção 4.1), adoptadas pelo projecto WHIPS (secção 4.1.1) e pelo projecto Squirrel (secção 4.1.2), e as abordagens orientadas a modelos (secção 4.2), introduzidas pelo projecto DWQ (secção 4.2.1), e adoptadas pelo projecto SIRIUS (secção 4.2.2), pelo projecto Arktos (secção 4.2.3) e por J Trujillo e S Luján-Mora (secção 4.2.4).

4.1 Abordagens de Manutenção das Vistas Materializadas

O problema do refrescamento dos RDW é inicialmente abordado na perspectiva da manutenção das vistas, entendido como o problema de computar as alterações ocorridas nas fontes de dados e actualizar as vistas materializadas, garantindo a sua consistência e a optimização da operação (Gupta e Mumick 1995: 3). Nesta perspectiva, o DW global é concebido como um conjunto de vistas materializadas definidas sobre as relações base nas fontes de dados (Widom 1995: 2).

Colocada na agenda científica dos SDW pela equipa de investigadores do projecto WHIPS (Gupta e Mumick 1995, Hammer et al. 1995, Wiener et al. 1996), esta abordagem seria também adoptada no projecto Squirrel (Zhou et al. 1995) e faria emergir diversas questões de pesquisa desenvolvidas por vários autores. Realçam-se os contributos de Roussopoulos e sua equipa da Universidade de Maryland, com o pioneiro projecto *Adaptative Database Management System* -ADMS- (Roussopoulos et al. 1995 e Roussopoulos 1998), e dos investigadores do Instituto Politécnico de ter, também nos EUA, que têm prosseguido esta abordagem e proposto novos algoritmos para optimizar a performance da manutenção incremental das vistas materializadas (Ding et al. 1999a e 1999b, Liu et al. 2002 e 2003).

4.1.1 Projecto WHIPS

O *framework* WHIPS foca na manutenção incremental das vistas materializadas[65] e propõe um conjunto de técnicas, algoritmos e componentes de arquitectura para a detecção e notificação das alterações ocorridas nas relações base e consequente propagação e actualização das vistas materializadas (Gupta e Mumick 1995, Labio e Garcia-Molina 1996, Zhuge et al. 1995, 1996, 1997 e 1998, Quass e Widom 1997, Garcia-Molina et al. 1998, Labio et al. 1997 e 1999, Gupta e Srivastava 1999).[66]

Dos contributos, destacam-se: (1) os algoritmos descritos por Zhuge et al. para manter os níveis desejados de consistência das vistas materializadas, quando ocorrem alterações nas respectivas relações base (1996, 1997 e 1998); (2) as técnicas de manutenção autónoma das vistas materializadas para garantir a sua actualização sem ter que aceder às relações base, optimizando a operação (Gupta et al. 1996, Garcia-Molina et al. 1998); (3) o desenvolvimento de componentes de arquitectura e a identificação e classificação das fontes de dados, de acordo com as capacidades que providenciam para ajudar a resolver o problema da detecção e notificação das alterações ocorridas nos seus conteúdos (Widom 1995: 27); (4) os algoritmos para calcular o diferencial de alterações, com base na comparação periódica entre sucessivos *snapshots* fornecidos off-line, quando as fontes de dados não oferecem capacidades para detectar as alterações relevantes ocorridas nos seus conteúdos (Labio e Garcia-Molina 1996).

Na arquitectura do sistema WHIPS, o componente central é o integrador que coordena a manutenção das vistas materializadas. As vistas são definidas pelo administrador do sistema, cabendo ao especificador das vistas verificar as definições das vistas, adicionando informação armazenada no repositório de metadados (sobre as fontes de dados, as relações base e os respectivos esquemas), e enviá-las ao integrador que notifica as fontes de dados e os gestores das vistas que gerem as vistas (ver Figura 4.1).[67]

---

[65] É o crescente interesse da comunidade científica e da indústria pelos SDW que levaram a um rejuvenescimento da área de estudo das vistas materializadas e à identificação e pesquisa das dificuldades que surgiram com a sua utilização em ambientes *warehouse*, ricos em informação analítica (ver Gupta e Mumick 1999a, 1999b, 1999c e 1999d).

[66] Para uma avaliação das técnicas de manutenção das vistas materializadas, ver Wang et al. (2000).

[67] O componente especificador das vistas e o respectivo *wrapper* estão associados às tarefas do administrador do sistema, no que refere à gestão da informação armazenada no repositório de metadados.

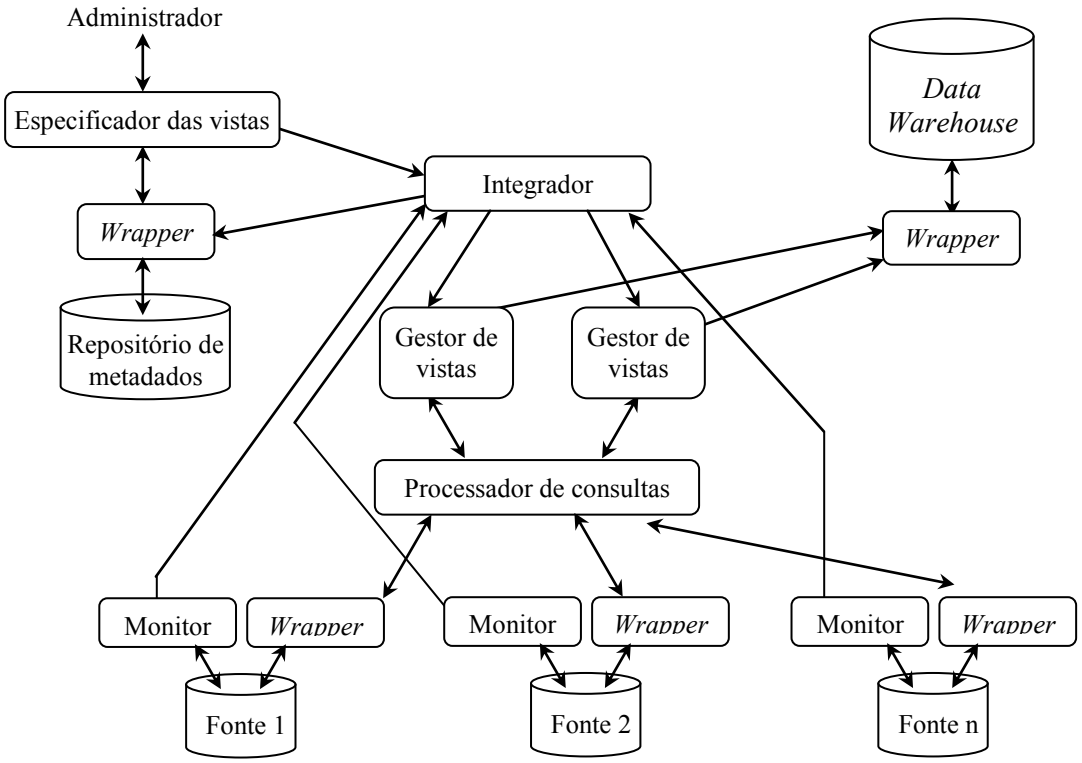


Figura 4.1 Arquitectura do WHIPS

Cada monitor detecta as alteraçõe*s* ocorridas nos dados da respectiva fonte e envia-as para o integrador que se encarrega de distribuir as alterações pelos respectivos gestores de vistas. Por sua vez, cada gestor de vista computa as correspondentes alterações à vista, utilizando o algoritmo definido para suportar a sua consistência. A actualização de uma vista faz-se através de consultas solicitadas ao processador de consultas que, por sua vez, solicita o *wrapper* de cada uma das fontes de dados envolvidas para a tradução de cada consulta na linguagem nativa da respectiva fonte de dados. Os resultados são então devolvidos ao gestor de vista que os combina e ajusta ao nível de consistência definido para a vista, para, de seguida, enviar as alterações para o *warehouse wrapper* encarregue de proceder à actualização das vistas materializadas armazenadas no *data warehouse* (Wiener et al. 1996: 3).

4.1.2 Projecto Squirrel

Também os investigadores do projecto Squirrel apresentam um *framework* que conjuga algoritmos, técnicas e componentes de arquitectura com o objectivo de propor uma solução para o problema da manutenção das vistas materializadas, embora com diferenças significativas face à abordagem WHIPS. Optam por uma abordagem híbrida

de integração das alterações ocorridas nas relações base, em que as vistas são classificadas como materializadas, virtuais e híbridas (Zhou et al. 1995, Hull e Zhou 1996), levantando-se aqui o problema da selecção de quais as vistas a materializar e questões como a materialização parcial ou total de uma vista (ver Gupta 1997 e Yang et al. 1997).

O *framework* baseia-se no conceito de mediador, um componente de software que suporta o processo de integração e um dado estado de consistência e de actualização das vistas (Hull e Zhou 1996). O mediador Squirrel inclui dois elementos fundamentais, o Plano de Decomposição das Vistas (PDV), um grafo que representa as interdependências entre as relações base e as vistas, e o conjunto de regras activas que suportam a manutenção incremental das vistas materializadas.

Na arquitectura do mediador Squirrel, o componente central é um repositório local que armazena as porções materializadas das vistas e outros dados auxiliares, o PDV e o conjunto de regras activas. O *update-queue* guarda as alterações detectadas nas fontes de dados e notificadas por capacidades oferecidas pelas fontes de dados, e envia-as ao processador incremental de alterações que, por sua vez, propaga essas alterações para as vistas materializadas, executando o respectivo algoritmo e de acordo com o PDV e as regras definidas para as vistas. O processador de atributos virtuais encarrega-se de construir as relações temporárias que suportam o acesso às porções virtuais das vistas, quando o processador de consultas assume que uma dada consulta não pode ser respondida apenas com as porções materializadas e precisa de recorrer às fontes para recolher os dados que não foram materializados no repositório local (ver Figura 4.2).

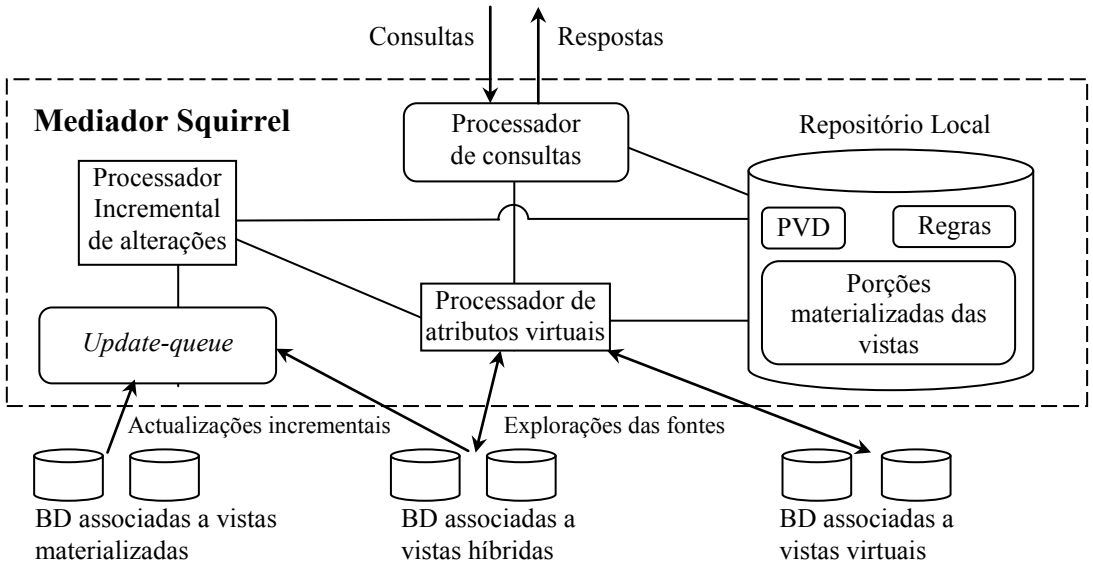


Figura 4.2 Arquitectura do mediador Squirrel

As fontes de dados são assumidas como bases de dados e classificadas de acordo com as suas capacidades activas[68] (Zhou et al. 1995: 34) e em função do tipo de vistas a que estão associadas (Hull e Zhou 1996: 485). No caso das fontes sem capacidades activas, cabe ao mediador explorar as fontes periodicamente (*pooling*) para detectar alterações e actualizar a informação replicada (Zhou et al. 1995: 34).[69]

4.2 Abordagens Orientadas a Modelos

São os contributos do *framework* de desenho e modelação providenciado pelo projecto DWQ que favorecem o desenvolvimento das abordagens orientadas a modelos, no domínio específico do processo de refrescamento dos RDW e no domínio alargado do processo de ETC, entendidos no âmbito do ciclo de vida de um projecto de SDW.[70] As abordagens centradas no desenho e modelação propõem modelos e metodologias de suporte, providenciando metadados passivos que se constituem como documentação para apoiar a especificação formal dos processos ou que permitem a especificação de metadados activos implementados em protótipos que suportam a modelação e execução dos processos.

As abordagens orientadas a modelos viriam a ser adoptadas pelos investigadores do projecto SIRIUS que focalizam no processo de refrescamento e pelos investigadores que centram a sua pesquisa no processo de ETC, dos quais se distinguem os investigadores do projecto Arktos e Luján-Mora e Trujillo. Embora a investigação não tenha revelado estudos no domínio estrito da modelação conceptual do processo de refrescamento, são contributos essenciais as raras propostas dos autores do domínio alargado do processo de ETC.

Os modelos são representações abstractas da realidade e utilizam um conjunto limitado de símbolos com um significado específico, procurando eliminar a ambiguidade e redundância geralmente associadas a uma descrição escrita e tirar partido

[68] As bases de dados activas são entendidas como SGDB com regras de produção, que fornecem um conjunto de mecanismos que permitem controlar a ocorrência de situações de interesse e a definição de um conjunto de acções a tomar (*event-condition-action rules*), incluindo o cumprimento de restrições de integridade, a manutenção de dados derivados, os *triggers* ou os *alerters* (Dayal et al. 1994).

[69] A este tipo de relação entre as fontes de dados e o sistema de integração (neste caso o mediador) está associada uma tarefa de extracção assente numa política *pull*, contrapondo-se à política *push* em que as fontes notificam e enviam as alterações ocorridas nos seus dados (Bouzeghoub e Peralta 2004: 61-2).

[70] Apesar de não serem do domínio estrito dos SDW, são também de referir os protótipos de ferramentas para limpeza de dados AJAX, desenvolvido no Institut National de Recherche en Informatique et en Automatique (Galhardas et al. 2000a e 2000b, Galhardas 2001: 97-126) e Potter's Wheel, desenvolvido na Universidade de Berkeley (Rahm e Hellerstein 2001).

da imagem como elemento de comunicação (Nunes e O'Neil 2003: 2). Constituem-se como *blueprints* que permitem representar, visualizar, documentar, especificar e construir um sistema, tornando possível comunicar o comportamento e a estrutura do sistema, visualizar e controlar a sua arquitectura, compreendê-lo melhor e gerir o risco (Booch et al. 2005: 5). A complexidade do processo de refrescamento torna os *blueprints* documentação fundamental para apoiar o desenho e a manutenção dos sistemas que suportam a sua execução (Vassiliadis 2005b e 2005c).

4.2.1 Projecto DWQ

No *framework* de desenho e desenvolvimento proposto pelo projecto DWQ constituem aspectos centrais e inovadores os factores de qualidade e a gestão dos SDW suportada por um repositório central de metadados. Os metadados são providenciados por modelos que capturam todos os aspectos do sistema, na perspectiva conceptual que representa a visão global dos objectos da organização, na perspectiva lógica constituída pelo esquema do DW global e os esquemas das fontes e na perspectiva dos componentes físicos que armazenam os dados das fontes e os dados *warehouse* (Jarke et al. 1999, Vassilidadis et al 2000d e 2001b).

No domínio específico do problema do refrescamento dos RDW, o *framework* oferece uma metodologia informal para guiar a definição do processo, assente numa sequência de quatro actividades, inseridas no ciclo de vida de um projecto de SDW: (1) a análise dos requisitos para identificar as vistas, as fontes de dados e os factores de qualidade; (2) o desenho conceptual que providencia uma primeira definição do processo como um cenário planeado de possíveis estratégias; (3) o desenho lógico que transforma os cenários conceptuais numa especificação formal, em termos do algoritmo principal, regras e respectivas semânticas de execução; (4) o desenho físico que implementa o algoritmo principal e as regras e respectivas semânticas. Acresce que o processo deve ser gerido por metadados activos que representam todos os dados resultantes dos três níveis de desenho e o mapeamento entre eles e ainda informações sobre a frequência e o intervalo adequados à sua execução (Bouzeghoub et al. 1997: 9-13).

Um contributo fundamental dos investigadores do projecto DWQ traduz-se na introdução do conceito de *workflow* no domínio do problema do refrescamento dos RDW, fazendo a ruptura com as abordagens tradicionais de manutenção das vistas materializadas.

O conceito de *workflow* diz respeito a uma actividade que envolve a execução coordenada de um conjunto de tarefas e constrangimentos associados, levada a cabo por entidades humanas e ou componentes de software. A sua especificação exige o recurso a linguagens apropriadas para descrever as tarefas e as relações entre elas e todos os aspectos associados que são relevantes para o controlo e coordenação da sua execução, incluindo a calendarização e as entidades envolvidas, como os sistemas e os dados de enquadramento ou o modelo da organização (Casati et al. 1995: 341). A sequência das tarefas deve ser, pois, adequada a essa estrutura do fluxo e suficientemente flexível para permitir um reordenamento quando se alteram aspectos da estrutura (Casati et al. 1996: 439).

Na literatura, estão disponíveis várias perspectivas e uma diversidade de domínios que utilizam o conceito, assim como inúmeras linguagens de especificação (ver Aalst 1997 e Aalst e Hofstede 2002 e 2003). A indústria também tem disponibilizado alguns standards que incluem metamodelos para especificar processos de *workflow* e linguagens para modelar objectos e processos em vários domínios, adequando-se ao ambiente *warehouse* o standard da *Workflow Management Coalition*[71], o *Open Information Model* e o *Data Transformations Elements* da *Metadata Coalition* que usa a UML (ver Vassiliadis 2000a: 3.3 e 2001b).

É neste sentido que o refrescamento dos RDW deixa de ser visto como o problema da materialização de vistas e sua manutenção e passa a ser conceptualmente representado como um *workflow*, cujas rotinas dependem dos produtos disponíveis para a extracção, limpeza e integração de dados e cuja coordenação dos eventos que iniciam as tarefas depende do domínio das aplicações envolvidas e dos requisitos de qualidade. Cada modelo de *workflow*, designado por cenário, deve ser adaptado aos requisitos de qualidade e constrangimentos específicos de um dado projecto de SDW (Bouzeghoub et al. 1999).

De facto, a abordagem de *workflow* permite decompor o processo de refrescamento em tarefas, ordená-las e organizá-las de acordo com cenários específicos de refrescamento, atendendo à evolução dos requisitos definidos pelos utilizadores, em termos de alterações das vistas e dos requisitos de qualidade, e à evolução dos constrangimentos impostos pelos repositórios de dados envolvidos. As tarefas ocorrem em quatro passos: (1) a preparação, com a tarefa de extracção dos dados alterados em

[71] Ver o WfMC *no sítio* http://www.wfmc.org.

cada uma das fontes, desde a última extracção (diferencial de alterações) e consequentes tarefas de limpeza e gestão do histórico, que pode ocorrer mais tarde; (2) a integração, com as tarefas de reconciliação das alterações extraídas das múltiplas fontes de dados e posterior carregamento no ODS; (3) a agregação, com as tarefas de recomputação incremental das alterações ao longo da hierarquia das vistas agregadas; (4) a fase de customização, com as tarefas de propagação dos dados sumariados para os *data marts* (ver Figura 4.3).

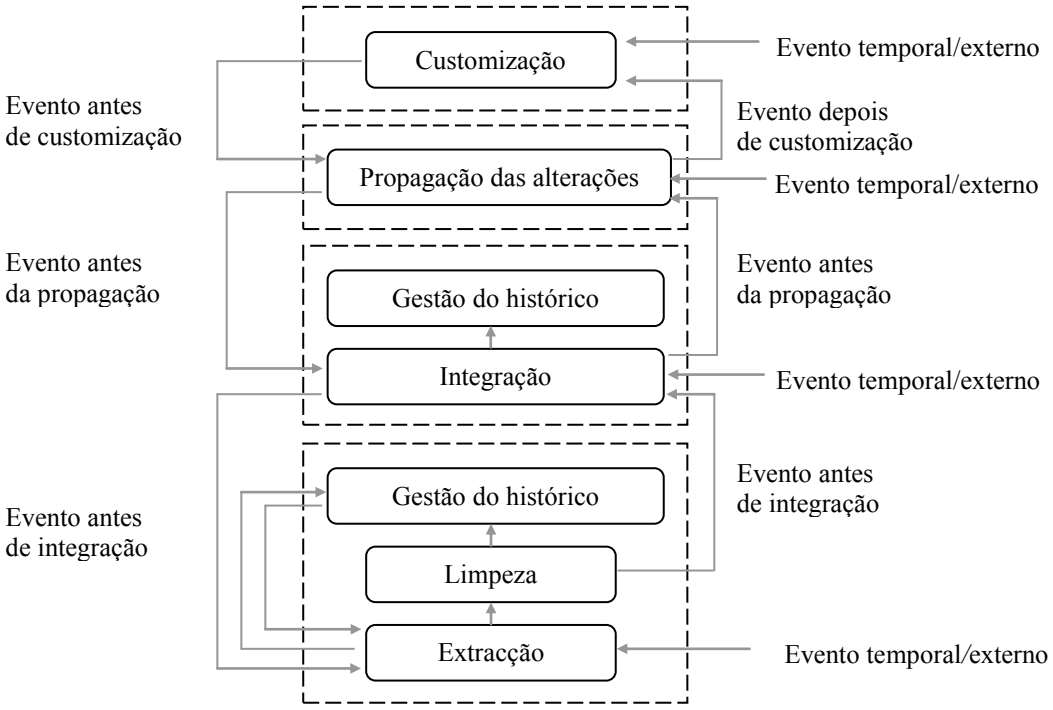


Figura 4.3 *Workflow* genérico para o processo de refrescamento

Para além da metodologia e da introdução do conceito de *workflow*, os investigadores propõem também as dimensões de qualidade dos dados que devem ser asseguradas e os respectivos de factores de qualidade que associam às decisões de desenho do processo e aos requisitos dos utilizadores (Bouzeghoub et al. 2003: 73-7).[72] Providenciam ainda um *toolkit* genérico baseado num modelo lógico de regras activas que admitem poder ser executadas, independentemente das linguagens de especificação das regras e dos SGBD envolvidos, e com a flexibilidade suficiente para garantir a

[72] Ver a proposta de um *framework* para a análise da idade dos dados (*data freshness*), um dos atributos mais importantes na qualidade dos dados, que inclui as métricas e uma taxionomia (Bouzeghoub e Peralta 2004).

evolução de quaisquer cenários de refrescamento (Bouzeghoub et al. 1998 e 2003: 77-85).

4.2.2 Projecto SIRIUS

Também no âmbito do projecto SIRIUS, os investigadores abordam o processo de refrescamento na perspectiva do *workflow*.

O principal contributo desta abordagem consiste na introdução de uma arquitectura flexível de *middleware* para suportar a modelação e execução de tarefas de refrescamento incremental, num nível intermédio entre as fontes de dados e o DW global, e que possa ser usada em SDW que envolvem fontes de dados heterogéneas, independentemente das tecnologias e SGBD utilizados para gerir o armazenamento e a manipulação dos dados *warehouse* (Gatziu et al. 1998, Vavouras et al. 1999a, 1999b, 2000, Vavouras 2002).

A modelação do processo de refrescamento é colocada no *framework* de desenho do SDW, isto é, os resultados do desenho conceptual e lógico do DW global e informação específica sobre as fontes de dados são usados na especificação do processo de refrescamento.

Um aspecto central desta abordagem consiste na utilização de um metamodelo que providencia um conjunto de construções para definir os metadados necessários à execução do processo de refrescamento, tais como a especificação do esquema global, a descrição das fontes de dados, a definição dos mapeamentos entre os esquemas das fontes de dados e o esquema global, incluindo as tarefas de refrescamento e informação relativa ao modo como devem ser tratadas as alterações que ocorrem nos dados das fontes e à calendarização e serialização da execução (Vavouras 2002).

O esquema global define uma representação global e uniforme das fontes de dados e o esquema de armazenamento define a estrutura dos dados tal como estão armazenados no DW global. Na arquitectura de esquemas proposta, os componentes responsáveis pela execução das várias tarefas de refrescamento, sumariadas como o Gestor do Refrescamento do *Data Warehouse* (GRDW), operam no topo do esquema global (ver Figura 4.4).

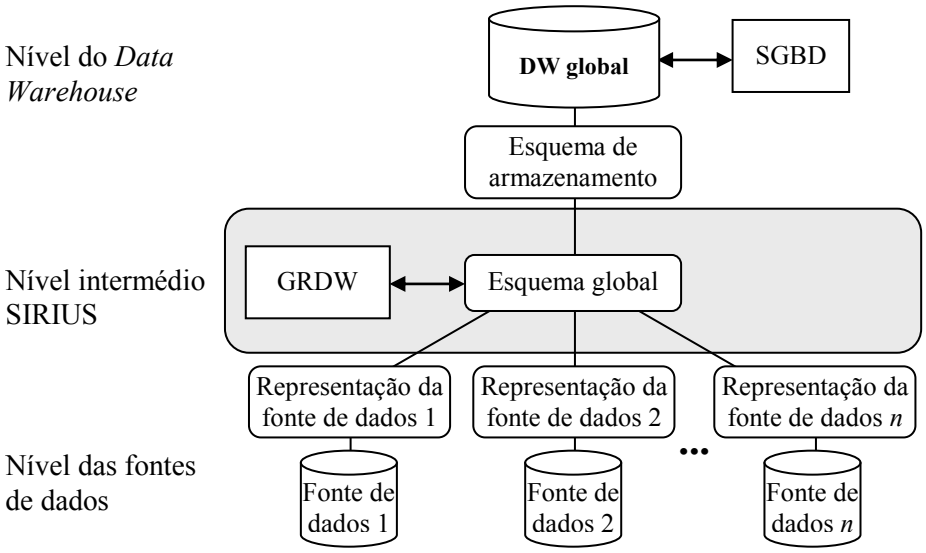


Figura 4.4 Arquitectura dos esquemas no SIRIUS

O protótipo que implementa a arquitectura de *midleware* é composto por: (1) um componente de modelação que disponibiliza os meios para o designer especificar e armazenar os metadados do processo de refrescamento (interface de utilizador e gestor de acesso ao repositório de metadados); (2) um componente de execução (GRDW) que utiliza os metadados para iniciar, controlar e monitorizar a execução do processo de refrescamento (coordenador), transformar as alterações nos dados das fontes em objectos do esquema global (gestor de objecto), implementar as tarefas associadas à gestão de chaves (gestor de chaves) e garantir o mapeamento semântico entre o esquema global e o esquema de armazenamento do DW global (*mapper* do esquema de armazenamento), persistindo os resultados intermédios da execução no repositório de execução.

O GRDW é complementado, no nível das fontes de dados, pelos monitores que detectam e notificam as alterações que ocorrem nos dados das fontes e pelos *wrappers* que escrevem essas alterações na área intermédia de armazenamento; no nível do *data warehouse*, pelo *wrapper* que carrega os dados *warehouse*, recorrendo às operações nativas de actualização do SGBD utilizado (ver Figura 4.5).

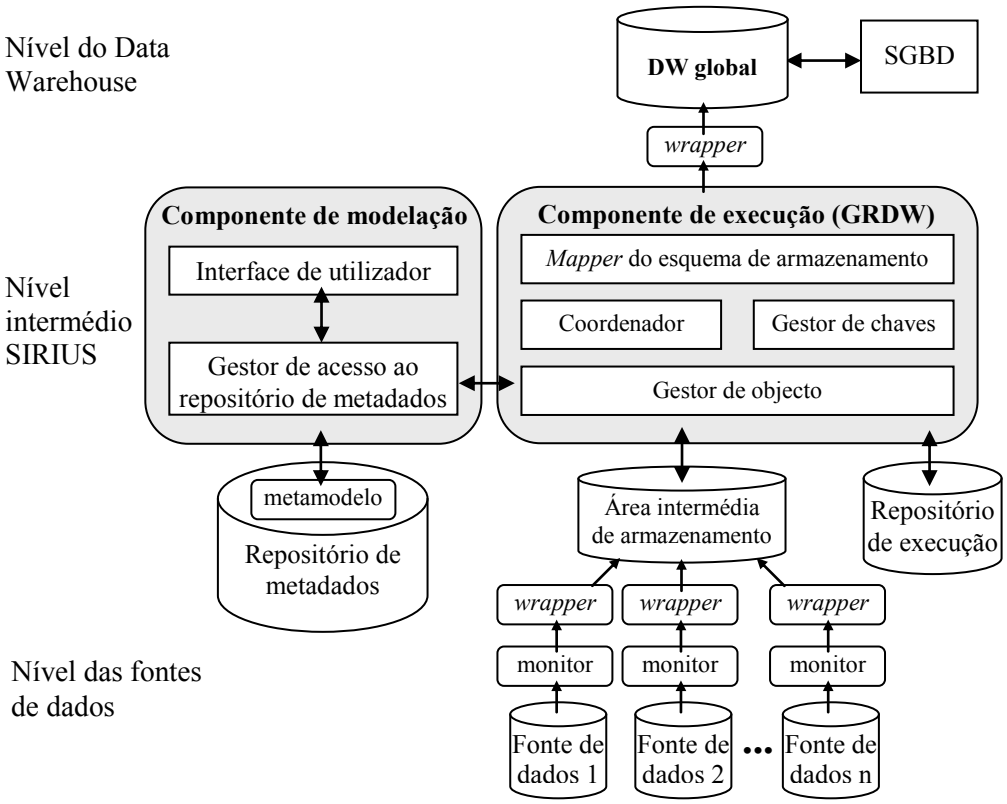


Figura 4.5 Protótipo que implementa a arquitectura de *midleware*

A abordagem proposta pelo projecto SIRIUS focaliza no nível intermédio, entre as fontes de dados e o DW global, para modelar as tarefas de limpeza de dados, de reconciliação e integração, de gestão de chaves e de gestão de histórico, enquanto que a tarefa de extracção é codificada em monitores e *wrappers*, sem o auxílio do metamodelo. Contudo, a abordagem possui os benefícios de utilizar uma linguagem de modelação orientada a objectos, estendida do standard da *Object Database Management Group* (e uma variante da *Object Interchange Format*), que pode facilitar a adaptação dos designers familiarizados com linguagens de modelação amplamente utilizadas, como a UML.

4.2.3 Projecto Arktos

O grupo de investigadores do projecto Arktos prossegue os experimentos no domínio dos formalismos de modelação dos processos operacionais como cenários de *workflows*, levados a cabo no âmbito do projecto europeu DWQ.

No domínio do processo de ETC, os investigadores desenvolvem um *framework* de modelação nos níveis conceptual, lógico e físico e que inclui uma arquitectura de metamodelo genérica, extensível e manuseável, os mapeamentos entre o modelo conceptual e o modelo lógico, um conjunto de algoritmos e a implementação de

protótipos, com o objectivo de facilitar, gerir e optimizar o desenho e execução do processo de ETC, ao longo do ciclo de vida de um SDW (Simitsis 2003 e 2005, Vassiliadis et al. 2003 e 2005a).

A arquitectura de metamodelo assenta numa estrutura de três camadas: (1) a camada superior de metamodelo que disponibiliza o modelo genérico que descreve todos os casos possíveis de aspectos de cenários de ETC; (2) a camada extensível e manuseável de *templates*, um subconjunto da camada de metamodelo que oferece construções específicas, isto é, instâncias especializadas dos casos genéricos, para modelar os aspectos mais frequentes de cenários de ETC e que possibilita ao designer introduzir, através de um mecanismo de especialização, os aspectos modelados repetidamente nos casos concretos dos seus projectos de SDW, tornando o metamodelo reutilizável; (3) a camada inferior de esquema que descreve o domínio de cenários específicos de ETC e cujas entidades são instâncias das duas camadas atrás referidas (Vassiliadis et al 2002a: 19-20, 2003: 537-40 e 2005a: 509-10).

Começa-se por sumariar os principais contributos nos níveis de desenho lógico e físico, para de seguida se tratar o nível conceptual.

Nos níveis lógico e físico, o primeiro esforço de pesquisa resulta no protótipo de ferramenta Arktos que oferece primitivas explícitas para modelar e executar as tarefas de limpeza e transformação numa dada sequência e alguns requisitos de qualidade de cenários práticos de ETC, disponibilizando três possibilidades para a descrição lógica dos aspectos a modelar: (1) um grafo suportado pelas primitivas de modelação ou uma de duas linguagens declarativas, (2) a XADL, uma variante da linguagem XML baseada no standard *Activity Definition Language*, que permite uma descrição mais verbal e de leitura mais fácil, e (3) a SADL (*Simple Activity Definition Language*), uma linguagem com uma sintaxe mais compacta que lembra a SQL e, por isso, mais adequada a designers experientes (Vassiliadis et al 2000a e 2001a).

É no âmbito da implementação do protótipo de ferramenta Arktos II que se introduzem melhorias significativas no *framework*: (1) o modelo lógico é reduzido a um grafo que permite descrever todos os casos de cenários de ETC nos seus aspectos estáticos e dinâmicos, dadas as características da arquitectura do metamodelo lógico; (2) a inclusão das semânticas do *workflow* e das operações de inserir, eliminar e alterar; (3) a utilização dos recursos de uma linguagem declarativa de programação de bases de dados (LDL++) para permitir descrever também as semânticas do *workflow* (4) a introdução de um algoritmo para suportar o *zooming in/out* nos vários níveis de detalhe

do grafo, de modo a lidar com a complexidade do nível de detalhe dos atributos; (5) a inclusão de métricas de qualidade; (6) o armazenamento dos cenários de ETC no repositório de metadados Arktos II, implementado num SGBD que facilita o processamento de consultas e permite extensões da ferramenta e integração com sistemas exteriores (Vassiliadis et al 2002b, 2003, 2005a e 2005b).

Por seu lado, o objectivo da modelação no nível conceptual é constituir documentação para apoiar as fases subsequentes de desenho, remetendo-se os outros aspectos do processo de ETC e as soluções técnicas da sua implementação para o nível lógico de desenho (Vassiliadis et al. 2002a: 14-5). O mapeamento entre o modelo conceptual e o modelo lógico é feito de modo semi-automático, cabendo a um algoritmo suportar a ordem de execução dos aspectos modelados (Simitsis 2005).

O nível conceptual de modelação é localizado na fase inicial do desenho de um SDW e diz respeito a duas tarefas que o designer executa em paralelo: (1) a recolha dos requisitos definidos pelos utilizadores; (2) a análise da estrutura e do conteúdo das fontes de dados e seus mapeamentos intencionais para o modelo comum do DW global (Vassiliadis et al. 2002a). Trata-se de representar, conceptualmente, o mapeamento dos atributos das fontes de dados para os correspondentes atributos das tabelas de dados do DW global e as transformações que devem ocorrer.

O *framework* de modelação conceptual oferece uma linguagem de especificação assente num conjunto de notações gráficas que permitem tratar os atributos como elementos de modelação de primeira classe. A decisão por notações próprias é a solução encontrada pelos investigadores para modelar neste nível de detalhe, com o argumento da inadequação da UML para modelar entidades de baixa granularidade como é o atributo (ver Apêndice 8.1.1).

Dadas as características da arquitectura do metamodelo conceptual, o designer descreve o domínio de cenários específicos de ETC na camada de esquema do metamodelo, instanciando as classes da camada genérica de metamodelo e as sub-classes especializadas oferecidas pela camada de *template*, onde pode incluir os padrões mais recorrentes nas suas actividades de modelação (ver Figura 4.6).

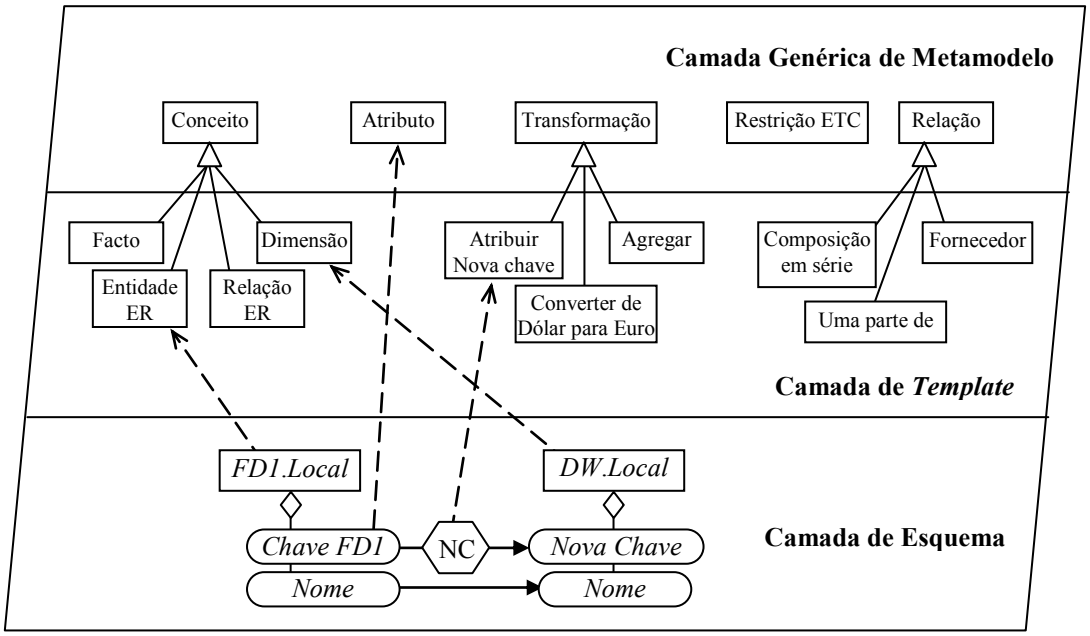


Figura 4.6 Arquitectura do metamodelo conceptual

O *framework* disponibiliza ainda um método para apoiar o designer (Simitsis e Vassiliadis 2003). Nas suas actividades de modelação conceptual, o designer deve executar um conjunto de quatro passos em sequência: (1) a identificação das fontes de dados; (2) a identificação das fontes de dados candidatas e candidatas activas; (3) o mapeamento dos atributos entre as fontes de dados e o DW global; (4) a anotação do diagrama com as restrições de execução.

4.2.4 J Trujillo e S Luján-Mora

Trujillo e Luján-Mora integram a sua proposta de modelação conceptual do processo de ETC num *framework* global e integrado de desenho e de uma metodologia de desenvolvimento de SDW, baseado na UML e no *Unified Software Development Process* (USDP), com o objectivo de reduzir o tempo de desenvolvimento, facilitar a gestão do repositório de metadados e do sistema e permitir ao designer estimar o impacto das alterações nos esquemas das fontes de dados no esquema do DW global. As principais vantagens deste *framework* flexível residem: (1) no recurso a uma linguagem standard e expressiva como a UML para o desenho conceptual, lógico e físico de todos os componentes do SDW; (2) na utilização dos diagramas da UML para os modelos das diferentes fases e níveis de desenho conceptual, lógico e físico, permitindo uma visão integrada do sistema em vários níveis de abstracção; (3) no uso de mecanismos de agrupamento (pacotes da UML) para lidar com a complexidade do domínio modelado e tornar possível o *zooming in*/*out* nos vários níveis de detalhe; (4) no processo de

engenharia baseado no USDP que especifica o método de desenvolvimento e interliga todos os diagramas (Luján-Mora e Trujillo 2003 e 2004).

O *framework* proposto permite distinguir cinco fases de desenho durante as quais são representados (1) os esquemas das fontes de dados; (2) os mapeamentos entre as fontes de dados e o DW global (integração); (3) o esquema do DW global; (4) os mapeamentos entre o DW global e os repositórios locais (customização); (5) o esquema dos repositórios locais (Luján-Mora et al. 2004). O *framework* permite o desenho nos níveis conceptual, lógico e físico, através de diferentes diagramas da UML e correspondentes conjuntos coerentes de extensões definidas para um propósito específico, os *profiles* (ver Tabela 4.1).

	Fonte de dados (FD)	**Integração**	***Data Warehouse* (DW)**	**Customização**	**Repositórios Locais (RC)**
Conceptual	Esq. conceptual FD Diagrama de classes UML standard	Mapeamento (entre FD e DW) Diagrama de classes *Data Mapping Profile*	Esq. conceptual DW Diagrama de classes UML standard	Mapeamento (entre DW e RC) Diagrama de classes *Data Mapping Profile*	Esq. conceptual RC Diagrama de classes UML standard *Multidimensional Profile*
Lógico	Esq. lógico FD Diagrama de classes Vários *profiles* de UML	Processo ETC Diagrama de classes *ETL Profile*	Esq. lógico DW Diagrama de classes Vários *profiles* de UML	Exportação Diagrama de classes *ETL Profile*	Esq. lógico RC Diagrama de classes Vários *profiles* de UML
Físico	Esq. físico FD Diagramas de componentes e instalação *Database deployment profile*	Instalação Diagrama de instalação *Database deployment profile*	Esq. físico DW Diagramas de componentes e instalação *Database deployment profile*	Instalação Diagrama de instalação *Database deployment profile*	Esq. físico RC Diagramas de componentes e instalação *Database deployment profile*

Tabela 4.1 *Framework* de desenho do SDW

O *framework* define um diagrama de classes (*ETL profile*) para representar o processo de ETC no nível lógico e um diagrama de classes (*data mapping profile*) para representar os mapeamentos entre as fontes de dados e o DW global e os mapeamentos entre o DW global e os repositórios locais.

Apesar de localizado no nível lógico do *framework*, o *ETL profile* é inicialmente apresentado para suportar a modelação conceptual das tarefas mais comuns, como a limpeza de dados e a reconciliação e integração de dados (Trujillo e Luján-Mora 2003). A modelação conceptual é situada na fase inicial do desenho de um SDW, podendo o designer definir uma visão global de todo o processo e deixar a sua especificação formal ao programador das bases de dados ou providenciar uma descrição detalhada das tarefas mais comuns.

O *ETL profile* permite decompor o desenho do processo de ETC num conjunto reduzido de mecanismos de extensão, de modo a gerir a complexidade do processo, bem como a integrar o desenho no *framework* global e integrado. O modelo disponibiliza um

conjunto de classes estereotipadas da UML que representam os vários mecanismos de ETC e define os ícones que podem substituir as representações standard das classes estereotipadas. As relações entre os mecanismos são representadas através da dependência da UML, podendo ainda anexar-se uma nota a cada mecanismo de ETC para explicar o seu funcionamento ou para definir os mapeamentos entre os atributos das fontes de dados e os atributos do DW global (ver Apêndice 8.2.1).

Por seu lado, o *data mapping profile* é localizado no nível conceptual do *framework* e disponibiliza mecanismos de extensão da UML que permitem tratar os atributos e suas inter-relações como elementos de modelação de primeira classe para representar os mapeamentos entre os atributos nas fontes de dados e os atributos no DW global, incluindo as transformações e restrições de execução (Luján-Mora et al. 2004). O diagrama é suficientemente flexível para permitir descrever particularidades do fluxo de dados e as relações entre os repositórios de dados envolvidos nos níveis de detalhe das bases de dados, do fluxo de dados, das tabelas e dos atributos e aproveitar o *zooming in*/*out* (ver Apêndice 8.2.2).

Para além dos modelos conceptuais, o *framework* de desenho do processo de ETC providencia algumas orientações metodológicas e passos a seguir pelo designer: (1) a selecção das fontes de dados para a extracção; (2) a transformação dos dados extraídos, podendo incluir a limpeza para remover erros e inconsistências; (3) a combinação dos dados; (4) a selecção dos repositórios a popular; (5) o mapeamento dos atributos das fontes de dados para os correspondentes atributos dos repositórios alvo; (6) o carregamento dos dados (Trujillo e Luján-Mora 2003: 309).

5 MODELAÇÃO DA TAREFA DE EXTRACÇÃO

Este capítulo trata a parte conceptual da definição do processo de refrescamento, localizando-se na fase inicial do desenho dos SDW e focalizando-se na tarefa de extracção de dados das fontes. Propõe um guião para apoiar a modelação conceptual da tarefa de extracção de dados, começando por definir-se os conceitos utilizados (secção 5.1) e por descrever-se o exemplo corrente que ilustra a aplicação desses conceitos em cenários específicos de extracção (secção 5.2). Apresenta-se a sequência de passos que devem ser seguidos pelo designer para modelar a tarefa de extracção de dados (secção 5.3) e providenciam-se as construções que permitem representar os dados relevantes que se extraem das fontes (subsecção 5.3.1) e as regras que permitem isolar e extrair os dados relevantes para a tomada de decisão (subsecção 5.3.2).

5.1 Conceitos Utilizados

O domínio da extracção de dados considerado compreende as fontes de dados, os requisitos de informação traduzidos nos dados relevantes e nas regras que permitem isolar e extrair esses dados para a tomada de decisão, em dados cenários de extracção.

A utilização de um conjunto de conceitos propostos pelo FRISCO mostra-se útil em duas perspectivas (Falkenberg et al. 1998).[73]

Primeiro, permite entender e organizar um conjunto de aspectos que fazem parte do modelo da extracção de dados, isto é, da concepção do domínio da extracção de dados que pode ser representada numa linguagem formal ou semi-formal. Apesar de se tratar de uma concepção específica de um dado domínio de refrescamento, o modelo da extracção de dados compreende aspectos organizados numa estrutura conceptual expressiva que se pode adequar a quaisquer cenários de extracção.

Segundo, o relatório propõe um *framework* consistente e pragmático de conceitos da área dos sistemas de informação, que surge da preocupação de uniformização dos termos utilizados pelos práticos e pela comunidade científica.

[73] O FRISCO é utilizado também por Pelozo, que trata a modelação conceptual dos requisitos recorrendo a uma linguagem orientada-a-objectos (2003).

O conjunto de conceitos assenta no tetraedro semiótico, distinguindo-se os conceitos básicos do mundo das concepções e os conceitos básicos do mundo das representações, nos níveis intencional e extensional (Falkenberg et al. 1998: 33-7).[74]

Parte-se do entendimento de **concepção** como uma construção mental produzida por um dado actor social e que traduz o modo como é interpretado um fenómeno percepcionado a partir da observação de um determinado domínio do mundo real. Por seu lado, r**epresentação** é entendida como a descrição de uma dada concepção, numa linguagem apropriada.

O actor social concebe o domínio percepcionado, decompondo-o em partes ou aspectos identificáveis a que se designa de **coisa**. O conjunto de todas as coisas consideradas pelo actor social é a sua concepção desse domínio.

Exemplo: "Os Lusíadas", "foi escrito por" e "Os Lusíadas foi escrito por Luís de Camões" são coisas de uma concepção de um dado domínio.

Os **predicados** são coisas que podem ser utilizadas para caracterizar ou qualificar outras coisas. As **coisas predicadas** são coisas caracterizadas ou qualificadas por um ou vários predicados.

Exemplo: "Luís de Camões" é uma coisa predicada caracterizada ou qualificada pelos predicados "é português", "é natural de" e "é o autor de".

A **relação** é uma coisa composta por uma ou mais coisas predicadas, cada uma delas associada a um predicado que caracteriza o papel da coisa predicada na relação.

Exemplo: "Luís de Camões é português" é uma relação composta pela coisa predicada "Luís de Camões" e pelo predicado "é português" que caracteriza essa coisa predicada na relação.

Uma **relação *n*-ária** é uma relação composta por *n* pares coisa predicada-predicado. As cardinalidade destes pares é usualmente pequena.[75] As relações podem também ser coisas predicadas noutras relações.

Exemplo: "Um dado paciente tem 18 anos" é uma relação unária composta pela coisa predicada "Um dado paciente" e pelo predicado "tem 18 anos". "Luís de Camões escreveu Os Lusíadas" e "Os Lusíadas foi escrito por Luís de Camões" são uma mesma relação binária. "foi escrito por" e "escreveu" são predicados da referida relação.

[74] Para uma definição formal destes termos ver Falkenberg et al. (1998: 92-100).

[75] No modelo representado, consideram-se apenas as relações unárias e as relações binárias.

A **coisa elementar** designa uma coisa que não é uma relação e que não é caracterizada ou qualificada pelo predicado especial "tem o elemento" e a **coisa composta** é uma coisa que não é elementar. Note-se que as coisas predicadas são consideradas elementares ou compostas, dependendo do contexto em que são utilizadas, mas que os predicados são sempre considerados coisas elementares.

Exemplo: O comandante de um navio pode considerar que o seu navio é uma coisa composta que contém múltiplos componentes e um conjunto de relações que interligam estes componentes. O Almirante que dirige uma frota de navios pode não estar interessado nos componentes individuais dos navios e considerar que os navios da sua frota são coisas predicadas elementares.

A noção de **tipo de coisas** assenta numa caracterização específica (predicados e regras) que se aplica a todas as coisas desse tipo, permitindo dizer-se que essas coisas são **instâncias** desse tipo. O conjunto de instâncias de um dado tipo de coisas constitui uma **população**. As **regras** regulam um conjunto não vazio de tipos de coisas determinando as populações permissíveis num contexto específico.

Exemplo: o actor social concebe o tipo de coisas que respeitam a caracterização específica "é um adulto". O conjunto de todos os adultos considerados pelo actor social constitui uma população. O referido actor concebe a regra "todos os adultos tem mais de 18 anos" para determinar a referida população.

O actor social pode conceber os tipos de coisas como caracterizações simples de coisas predicadas elementares ou compostas (relações) e pode combinar as caracterizações simples para formar caracterizações mais complexas através das **junções binárias** de união, intersecção ou diferença.

Exemplo: um dado actor social concebe o tipo de coisas predicadas elementares que respeitam a caracterização "é uma pessoa" e que participam em relações unárias que contêm os predicados "chama-se (um dado nome)" e "nasceu no dia (uma dada data)". Os tipos de coisas predicadas que respeitam as caracterizações referidas por "é um homem" e "é uma mulher" são combinados para formar o tipo de coisas predicadas que respeitam a caracterização referida por "é um Homem OU é uma Mulher" através da junção binária de união.

No **nível intencional**, as coisas são organizadas num conjunto de tipos e de regras que regulam esses tipos. No **nível extensional**, as características específicas de cada tipo de coisas são estendidas às correspondentes populações.

Um **modelo** é uma concepção que pretende ser clara, precisa e não ambígua de um domínio específico e pode ser representado numa linguagem formal ou semi-formal apropriada. O **modelo intencional** é a parte do modelo que compreende as possibilidades e as necessidades de um dado domínio, isto é, os tipos de coisas e as regras. O **modelo extensional** é a parte do modelo que contém a população específica de cada um dos tipos de coisas do correspondente modelo intencional.

Com base nos conceitos apresentados, o modelo intencional da extracção de dados é organizado num conjunto de tipos de coisas e de regras:

1. Os tipos de coisas descritas pelas fontes de dados;
2. Os tipos de coisas relevantes definidos a partir dos requisitos de informação;
3. As regras que regulam os tipos de coisas relevantes e o modo como as coisas desses tipos são extraídas (técnicas de extracção incremental ou total).

O conjunto de tipos de coisas e de regras é representado, conceptualmente, nos *diagramas* da UML.[76]

A decisão pela UML deve-se ao facto de se tratar de uma linguagem standard *de facto* que permite visualizar, especificar, construir e documentar os artefactos de uma grande variedade de sistemas (Booch et al. 2005: 14 e ss.). Por outro lado, as linguagens orientadas a objectos, como a UML, têm a vantagem de utilizar uma noção de objecto que facilita o encapsulamento dos aspectos estruturais e de comportamento e que se aproxima do modo como o actor social concebe o domínio observado (Pelozo 2003: 17).

Os tipos de coisas descritos pelas diferentes fontes de dados são representados nos *diagramas de classes*, habitualmente utilizados no desenho de bases de dados.[77] Por esse motivo, não são considerados os conflitos de heterogeneidade decorrentes da utilização de diferentes linguagens de modelação, seguindo de perto as propostas de autores como Vassilidadis et al. (2002a) e Trujillo e Luján-Mora (2003).

Por outro lado, os tipos de coisas relevantes e as regras associadas são representados nos *diagramas de estruturas compostas* (OMG 2005: 157-88).

A decisão pelos *diagramas de estruturas compostas* deve-se, por um lado, ao facto de providenciarem os mecanismos de composição que permitem representar tipos de coisas compostas cujas partes têm características (predicados e as regras) que são

[76] A partir deste ponto, os termos introduzidos no contexto da linguagem UML estão em *itálico*.

[77] Ver, por exemplo, Booch et al. (2005: 106 e ss.).

específicas do contexto dos respectivos tipos.[78] Por outro lado, permitem articular os elementos dos *diagramas de classes* utilizados para representar os tipos de coisas descritas pelas fontes de dados com os elementos dos *diagramas de estruturas compostas* utilizados para representar os tipos de coisas relevantes e as regras associadas.

Admite-se que os diagramas utilizados possam ser complementados com os *diagramas de comportamento* da UML para capturar os aspectos dinâmicos da extracção, particularmente, as mensagens transferidas entre as fontes de dados e os componentes de extracção (por exemplo, os *wrappers*).

Ao longo da apresentação do guião, estes conceitos, bem como as construções dos diagramas, são clarificados através da sua aplicação no exemplo corrente.

5.2 Exemplo Corrente

O Instituto de Investigação Animal (IIA) pretende construir um SDW a partir da integração de duas fontes de dados independentes (FD1 e FD2), de modo a publicar informação relevante para o apoio à tomada de decisão nos assuntos relacionados com a investigação corrente e com a evolução dos animais em observação. Os conteúdos disponibilizados nos RDW devem ser regularmente refrescados para reflectirem as alterações que ocorrem nos conteúdos das fontes de dados que lhes servem de base.

O modelo intencional da extracção de dados, organizado com base nos conceitos propostos pelo FRISCO (Falkenberg et al. 1998), permite descrever o conjunto de tipos de coisas e de regras a representar.

Os dados da FD1 e da FD2 documentam as actividades que a equipa de estudantes e a equipa de investigadores desenvolvem nos ambientes Parque Zoológico e Selva, respectivamente. Os modelos intencionais da FD1 e FD2 são representados pelos *pacotes* FD1 e FD2, respectivamente (ver Figura 5.1).[79]

[78] Ver exemplos de utilização destes mecanismos em Bock (2004), Booch et al (2005: 203-4) e OMG (2005: 179-81).

[79] Ver Blaha e Premerlani para a modelação e desenho orientado a objectos de bases de dados (1998).

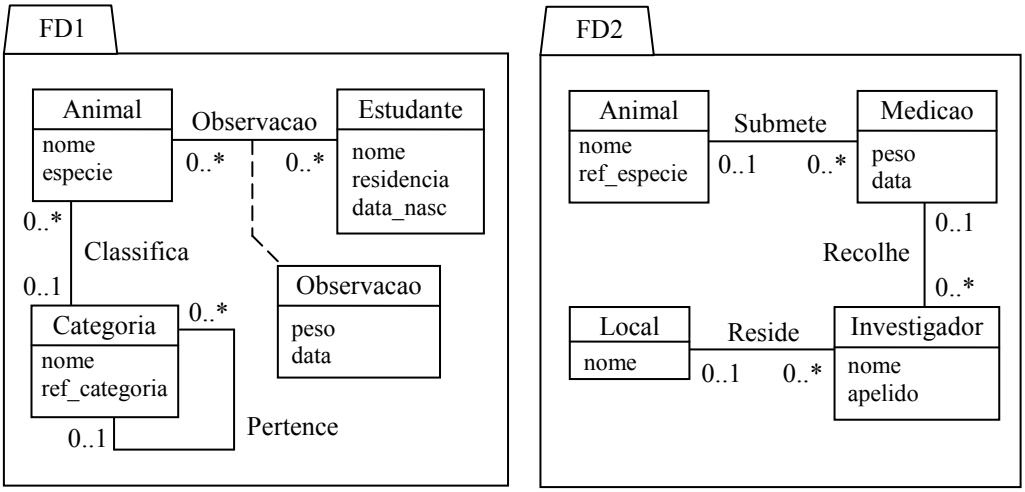


Figura 5.1 Modelos intencionais das fontes de dados, em UML

Os tipos de coisas e as regras representadas nos *pacotes* FD1 e FD2 são os seguintes:

1. **FD1**

1.1. A *classe* Animal descreve os animais observados, incluindo o nome do animal e o nome da espécie a que pertence (*atributos* nome e especie, respectivamente);

1.2. A *classe* Categoria descreve as categorias da hierarquia taxionómica utilizada para classificar os animais, concretamente, o nome e a referência da categoria (*atributos* nome e ref_categoria, respectivamente). As referidas categorias compreendem as espécies, os géneros e as famílias;

1.3. A *classe* Estudante descreve os estudantes responsáveis por recolher as observações, incluindo o nome do estudante, o local de residência e a data de nascimento (*atributos* nome, residencia e data_nasc, respectivamente);

1.4. A *classe associativa* Observacao descreve as observações recolhidas, concretamente, o peso do animal e a data de recolha; indica que um dado estudante pode observar vários animais (*multiplicidade* 0..*) e que um dado animal pode ser observado por vários estudantes (*multiplicidade* 0..*);

1.5. A *associação* Classifica descreve que um dado animal pode ser classificado com uma só categoria taxionómica (*multiplicidade* 0..1) e que uma dada categoria taxionómica pode ser utilizada para classificar vários animais (*multiplicidade* 0..*);

1.6. A *associação* Pertence descreve que uma dada categoria taxionómica pode pertencer a uma só categoria taxionómica de nível superior (*multiplicidade* 0..1) e que uma dada categoria taxionómica pode possuir uma ou várias categorias taxionómicas de nível inferior (*multiplicidade* 0..*).

2. **FD2**

2.1. A *classe* Animal descreve os animais observados, incluindo o nome do animal (*atributo* nome) e a referência que identifica univocamente a sua espécie (*atributo* ref_especie);

2.2. A *classe* Investigador descreve os investigadores responsáveis por recolher as medições, concretamente, o nome (*atributo* nome) e o apelido (*atributo* apelido);

2.3. A *classe* Local descreve os locais de residência dos investigadores (*atributo* nome);

2.4. A *classe* Medicao descreve as medições recolhidas, concretamente, o peso do animal e a data de recolha (*atributos* peso e data, respectivamente);

2.5. A *associação* Submete descreve que um dado animal pode ser submetido a várias medições (*multiplicidade* 0..*) e que uma dada medição pode ser submetida a um só animal (*multiplicidade* 0..1);

2.6. A *associação* Recolhe descreve que um dado investigador pode recolher várias medições (*multiplicidade* 0..*) e que uma dada medição pode ser recolhida por um só investigador (*multiplicidade* 0..1);

2.7. A *associação* Reside descreve que um dado investigador pode residir num só local (*multiplicidade* 0..1) e que num dado local podem residir vários investigadores (*multiplicidade* 0..*).

Na fase de levantamento do conjunto de requisitos exigidos para o desenho do SDW, foram tidos em conta os requisitos de informação que detalham as necessidades de informação dos utilizadores, permitindo identificar os tipos de coisas relevantes, atendendo aos modelos FD1 e FD2 (ver Tabela 5.2).[80]

1. Tipo "Animal do parque"

Coisa predicada: o animal (do modelo FD1).

Predicados: "chama-se (dado nome)", "classificado por (dada espécie)" e "vive (no parque)", "pertence a (dada categoria)" e "observado por (dados estudantes)".

2. Tipo "Animal da selva"

Coisa predicada: a relação composta pelo animal (do modelo FD2) e pela respectiva categoria taxionómica (do modelo FD1). Os predicados desta relação não existem nos modelos das fontes de dados.

Predicados: "chama-se (dado nome)", "classificado por (dada espécie)" e "vive (na selva)" e "submete-se a (dadas medições)".

3. Tipo "Animal"

Coisa predicada: o animal do parque OU o animal da selva.

Predicados: "chama-se (dado nome)", "classificado por (dada espécie)" e "vive (na selva ou no parque)".

[80]Para uma classificação dos requisitos necessários à construção de um SDW ver, por exemplo, Bruckner et al (2003).

4. Tipo "Observação"
Coisa predicada: a observação (do modelo FD1) OU a medição (do modelo FD2).
Predicados: "tem o peso (dado peso)", "efectuada no dia (dada data da recolha)", "recolhida por (dado estudante ou dado investigador)" e "submetida a (dado animal da selva ou dado animal do parque)".
5. Tipo "Estudante"
Coisa predicada: o estudante (do modelo FD2).
Predicados: "chama-se (dado nome)", "vive em (dado local de residência)", "pertence à equipa (estudante)" e "observa (dados animais)".
6. Tipo "Investigador"
Coisa predicada: a relação composta pelo investigador e pelo local onde reside (do modelo FD1).
Predicados: "chama-se (dado nome)", "vive em (dado local de residência)", "pertence à equipa (investigador)" e "recolhe (dadas medições)".
7. Tipo "Observador"
Coisa predicada: o estudante OU o investigador.
Predicados: "chama-se (dado nome)", "vive em (dado local de residência)" e "pertence à equipa (dada equipa)" e "recolhe (dadas observações ou dadas medições)".
8. Tipo "Hierarquia"
Coisa predicada: a coisa composta por 3 categorias taxionómicas (do modelo FD1).
Predicados: "categoriza a espécie (dada espécie)", "categoriza o género (dado género)", "categoriza a família (dada família)" e "classifica os animais (dados animais)".

Tabela 5.1 Tipos de coisas relevantes

Os requisitos de informação permitiram identificar as regras que regulam os tipos de coisas relevantes (ver Tabela 5.2), e as regras que regulam o modo como as coisas desses tipos são extraídas (ver Tabela 5.3).

1. Regra "O local de residência dos investigadores é opcional"
Aplica-se às relações que contêm as coisas predicadas "Investigador" e "Local" e os predicados "Reside" (do modelo FD2).
2. Regra "Os estudantes recolheram, pelo menos, uma observação"
Aplica-se às relações que contêm as coisas predicadas "Estudante", "Animal" e "Observação" (do modelo FD1).
3. Regra "Os investigadores recolheram, pelo menos, uma medição"
Aplica-se às relações que contêm as coisas predicadas "Investigador" e "Medição" e os predicados "Recolhe" (do modelo FD2).
4. Regra "Os estudantes têm, pelo menos, 18 anos de idade"
Aplica-se às coisas predicadas "Estudante" (do modelo FD1).

5. Regra "Os animais da selva podem referir uma categoria taxionómica (opcional)"
Aplica-se às relações que contém as coisas predicadas "Animal" e "Categoria" dos modelos FD2 e FD1, respectivamente.
6. Regra "O nome da espécie de um dado animal do parque é idêntico ao nome da categoria taxionómica que o classifica"
Aplica-se às relações que contém as coisas predicadas "Animal" e "Categoria" do modelo FD1. O valor da característica "especie" dos animais deve ser idêntico ao valor da característica "nome" das categorias que os classificam.
7. Regra "A categoria taxionómica dos animais da selva é identificada pela referência da espécie do animal"
Aplica-se às relações que contém as coisas predicadas "Animal" e "Categoria" dos modelos FD2 e FD1, respectivamente. O valor da característica "ref_especie" dos animais deve ser idêntico ao valor da característica "ref_categoria" das categorias que os classificam.

Tabela 5.2 Regras que regulam os tipos de coisas relevantes

1. Técnicas de extracção incremental
Os dados relativos às coisas dos tipos "Observador" (estudantes ou investigadores), "Animais" (do parque ou da selva) e "hierarquias" devem ser extraídos recorrendo a uma técnica de extracção incremental.
2. Técnicas de extracção total
Os dados relativos às coisas do tipo "Observações" (observações ou medições) devem ser extraídos recorrendo a uma técnica de extracção total.

Tabela 5.3 Modo como as coisas relevantes são extraídas

5.3 Guião de Modelação Conceptual da Tarefa de Extracção

Na fase inicial do desenho de um SDW, o designer leva a cabo duas actividades essenciais: (1) a análise da estrutura e conteúdos das fontes de dados identificadas e o seu mapeamento intencional para o modelo do DW alvo; (3) a identificação dos requisitos dos utilizadores (Vassialiadis et al. 2002a: 14-5).

Nessa fase, os modelos conceptuais constituem documentação essencial que facilita a comunicação entre os *stakeholders* e apoia o designer na especificação das tarefas de refrescamento, sem procurar providenciar uma solução técnica para a sua execução (Trujillo e Luján-Mora 2003, Vassiliadis 2005a).

O guião proposto conduz a actividade de modelação do designer, em 2 passos: (1) modelar os tipos de coisas relevantes para a tomada de decisão; (2) modelar as

regras que regulam os tipos de coisas relevantes e o modo como as coisas desses tipos são extraídas (técnicas de extracção incremental ou total).

Antes de iniciar esta actividade, o designer deve criar um novo *pacote* para organizar as construções utilizadas na modelação conceptual da tarefa de extracção de dados. Nos cenários ilustrados pelo exemplo corrente é criado o *pacote* ME.

5.3.1 Modelar os Tipos de Coisas Relevantes

Neste passo, o designer deve utilizar as *classes* «*extrair*» para modelar os tipos de coisas relevantes identificados durante a fase de levantamento de requisitos, concretamente, descrevendo as coisas predicadas e os respectivos predicados.

> **Definição 5.1** ***Classe* «*extrair*»** - a *classe* «*extrair*» representa um tipo de coisas relevantes, descrevendo um conjunto de coisas predicadas que respeitam uma mesma caracterização (predicados e regras). O *nome* da *classe* «*extrair*» designa o tipo de coisas relevantes.[81]

A *classe* «*extrair*» possui uma *estrutura interna* que fornece o contexto e que actua como um *namespace*[82] para os elementos que descrevem as coisas predicadas de um dado tipo, possibilitando a utilização de caracterizações específicas (predicados e regras) que apenas se aplicam às coisas predicadas no contexto desse tipo.

5.3.1.1 Modelar as Coisas Predicadas

O designer deve modelar as coisas predicadas na *estrutura interna* das *classes* «*extrair*». Recorde-se que as coisas predicadas podem ser (1) coisas elementares, (2) coisas compostas (relações) ou (3) coisas formadas a partir de junções binárias de união, intersecção ou diferença.

O designer deve utilizar as *propriedades* para modelar as coisas elementares de um dado tipo.

---

[81] A *classe* «*extrair*» é uma instância da metaclasse *Class* do *pacote StructuredClasses* (utilizada no *diagrama de estruturas compostas*) que estende a metaclasse *Class* do *pacote Kernel* (utilizada no *diagrama de classes*) da UML 2.0, herdando as características das metaclasses *EncapsulatedClassifier* e *StructuredClassifier*. Ver OMG para uma descrição detalhada destas metaclasses (2005: 157-88).

[82] *Namespace* é um elemento que pode conter outros elementos. Cada elemento pode pertencer a um único *namespace* e pode ser identificado por um nome.

Definição 5.2 ***Propriedade*** - a *propriedade* representa uma coisa elementar de um dado tipo. O *tipo* da *propriedade* indica a *classe* que descreve as coisas elementares desse tipo.

No exemplo corrente, utiliza-se a *estrutura interna* da *classe* «*extrair*» AnimalParque para modelar as coisas predicadas do tipo "Animal do parque". Utiliza-se a *propriedade* do *tipo* Animal (do FD1) para indicar que as referidas coisas predicadas são coisas elementares descritas pela *classe* Animal do *pacote* FD1 (ver Figura 5.2).

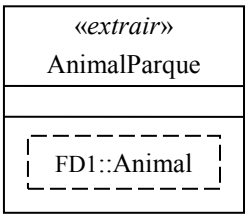


Figura 5.2 Modelar as coisas elementares

O designer deve indicar os *tipos* das *propriedades* utilizando os *prefixos* que permitem identificar univocamente as *classes* especificadas nos diferentes *pacotes*.[83] Os *prefixos* podem ser utilizados recorrendo à notação "*pacote*::*classe*" definida pela UML. Propõe-se, contudo, a utilização de uma notação alternativa, especificando o referido *prefixo* no canto superior esquerdo das *propriedades*.[84]

No exemplo corrente, utiliza-se a notação alternativa para especificar o *prefixo* FD1 que permite identificar univocamente a *classe* Animal do *pacote* FD1 (ver Figura 5.3).. Note-se que existe uma *classe* Animal em ambos os *pacotes* FD1 e FD2 (ver Figura 5.1).

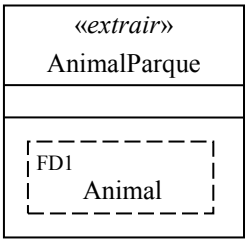


Figura 5.3 Notação alternativa do prefixo nas *propriedades* das *classes* «*extrair*»

[83] Os *prefixos* podem ser utilizados para indicar *classes* especificadas em *pacotes* que estão contidos noutros *pacotes*, com o separador "::" entre os *pacotes*. Ver OMG (2005).

[84] A notação alternativa deve ser entendida como uma opção de apresentação que não acrescenta qualquer significado aos diagramas, mas que pode ser visualmente apelativa por distinguir o *prefixo* e o nome da *classe* e por destacar o nome da *classe*.

O designer deve utilizar as *propriedades* interligadas por *conectores* para modelar as coisas compostas (relações) de um dado tipo.

Definição 5.3 ***Conector*** - o *conector* representa um predicado de uma relação binária de um dado tipo. O *tipo* do *conector* indica a *associação* que descreve os referidos predicados nos modelos das fontes de dados.

No exemplo corrente, utiliza-se a *estrutura interna* da *classe* «*extrair*» Investigador para modelar as coisas predicadas do tipo "Investigador". Utilizam-se as *propriedades* do *tipo* Investigador (do FD2) e Local (do FD2) e o *conector* do *tipo* Reside (do FD2) para indicar que as referidas coisas predicadas são relações compostas pelos investigadores que residem num dado local. "Os investigadores" e "os locais" são descritos pelas *classes* Investigador e Local do *pacote* FD2, respectivamente, e os predicados "reside" são descritos pela *associação* Reside do *pacote* FD2 (ver Figura 5.4).

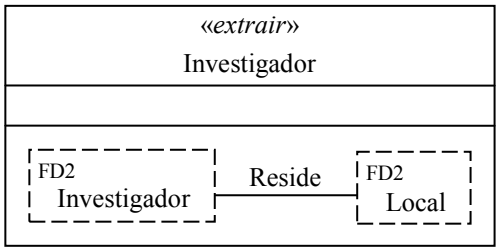


Figura 5.4 Modelar as relações

O designer deve omitir o *tipo* do *conector* para indicar que os predicados das coisas compostas não existem nos modelos das fontes de dados.

No exemplo corrente, utiliza-se a *estrutura interna* da *classe* «*extrair*» AnimalSelva para modelar as coisas predicadas do tipo "Animal da selva". Utilizam-se as *propriedades* do *tipo* Animal (do FD2) e Categoria (do FD1) e o *conector* não tipificado para indicar que as referidas coisas predicadas são relações compostas pelos animais que se classificam numa dada categoria taxionómica. "Os animais" são descritos pela *classe* Animal do *pacote* FD2 e "as categorias taxionómicas" são descritas pela *classe* Categoria do *pacote* FD1. Omite-se o tipo do *conector* para indicar que os predicados das referidas relações não existem nos modelos das fontes de dados (ver Figura 5.5).

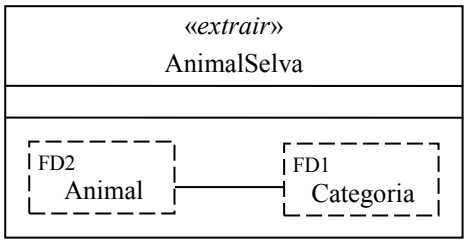


Figura 5.5 Modelar os predicados que não existem nos modelos das fontes de dados

O designer pode utilizar os *papéis* das *propriedades* para designar univocamente as coisas elementares na *estrutura interna* das *classes* «*extrair*», seguindo o formato "*papel*: *tipo*". Os *papéis* são obrigatórios nas relações compostas por múltiplas coisas elementares do mesmo tipo.

No exemplo corrente, utiliza-se a *estrutura interna* da *classe* «*extrair*» Hierarquia para modelar as coisas predicadas do tipo "Hierarquia". Utilizam-se as *propriedades* do *tipo* Categoria (do FD1) e os *conectores* do *tipo* Pertence (do FD1) para indicar que as referidas coisas predicadas são relações compostas pelas categorias das espécies, dos géneros e das famílias, interligadas por associações taxionómicas. "As espécies", "os géneros" e "as famílias" são coisas elementares descritas pela *classe* Categoria do *pacote* FD1 e os predicados "pertence" das associações taxionómicas são descritas pela *associação* Pertence do *pacote* FD1. Utilizam-se os *papéis* das *propriedades* Esp, Gnr e Fml para designar univocamente as categorias das espécies, dos géneros e das famílias, respectivamente (ver Figura 5.6).

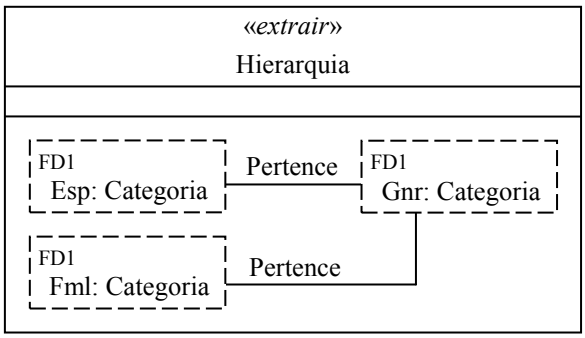


Figura 5.6 Modelar as relações com coisas elementares do mesmo tipo

Nota 5.1 As *associações* que descrevem os predicados dos modelos das fontes de dados podem ter uma *agregação* ou *composição* num dos seus *extremos*. A *multiplicidade* junto ao *extremo composto* é usualmente 0..1 ou 1..1.

O designer deve utilizar as *propriedades* e os *conectores* «*união*», «*intersecção*» e «*diferença*» para modelar as coisas predicadas formadas através das junções binárias de união, intersecção e diferença, respectivamente.

Definição 5.4 ***Conector* «*união*»** - duas *propriedades* dos *tipos* A e B interligadas por um *conector* «*união*» representam uma coisa predicada que é do *tipo* A ou é do *tipo* B.

Definição 5.5 ***Conector* «*intersecção*»** - duas *propriedades* dos *tipos* A e B interligadas por um *conector* «*intersecção*» representam uma coisa predicada que é do *tipo* A e é do *tipo* B.

Definição 5.6 ***Conector* «*diferença*»** - duas *propriedades* dos *tipos* A e B interligadas por um *conector* «*diferença*» representam uma coisa predicada que é do *tipo* A e que não é do *tipo* B (*tipo* de exclusão). O *extremo* do *conector* junto à *propriedade* do *tipo* de exclusão deve ser assinalado com o símbolo "-".

No exemplo corrente, utiliza-se a *estrutura interna* da *classe* «*extrair*» "Animal" para modelar as coisas predicadas do tipo "Animal". Utiliza-se o *conector* «*união*» entre as *propriedades* do *tipo* AnimalSelva (do ME) e AnimalParque (do ME) para indicar que as referidas coisas predicadas são coisas do tipo "Animal da Selva" ou coisas do tipo "Animal do Parque". As coisas do tipo "Animal da Selva" são descritas pela *classe* AnimalSelva do *pacote* ME e as coisas do tipo "Animal do Parque" são descritas pela *classe* AnimalParque do *pacote* ME (ver Figura 4.7).

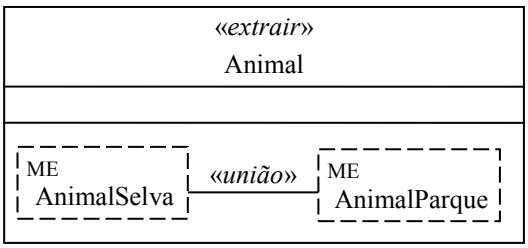


Figura 5.7 Modelar as coisas predicadas formadas por junções binárias

As *classes* que descrevem as coisas que participam nas junções binárias têm que ter *atributos* comuns, isto é, o *atributo* de ordem *n* de cada uma das *classes* tem que referir-se a um mesmo predicado, para todo *n*.

No exemplo corrente, as *classes* AnimalSelva e AnimalParque que descrevem as coisas que participam na união têm os *atributos* comuns na ordem: 1º nome, 2º espécie e 3º ambiente (ver os *atributos* na subsecção seguinte).

Nota 5.2 Quando uma dada *classe* tem *atributos* não comuns e/ou a ordem dos *atributos* é diferente, o designer deve criar uma nova *classe* «*extrair*», com uma *propriedade* do tipo da *classe* inicial e com os *atributos* comuns, devidamente ordenados.

5.3.1.2 Modelar os Predicados das Relações Unárias

O designer deve utilizar os *atributos* das *classes* «*extrair*» para modelar os predicados das relações unárias.

Definição 5.7 ***Atributo*** - o *atributo* da *classe* «*extrair*» representa um predicado que caracteriza as coisas predicadas numa relação unária. O *nome* do *atributo* designa a referida característica e o *tipo* do *atributo* indica o seu domínio de valores (real, inteiro, booleano ou cadeia de caracteres).[85]

No exemplo corrente, utilizam-se os *atributos* da *classe* «*extrair*» AnimalSelva para modelar os predicados que permitem caracterizar as coisas predicadas do tipo "Animal da Selva". Utilizam-se os *nomes* dos *atributos* nome, especie e ambiente para designar as características atribuídas pelos predicados "chama-se (um dado nome)", "classificado por (uma dada espécie)" e "vive (na selva)", respectivamente. Utiliza-se o *tipo de dados string* para indicar que o domínio de valores dos referidos atributos é cadeia de caracteres (ver Figura 5.8).

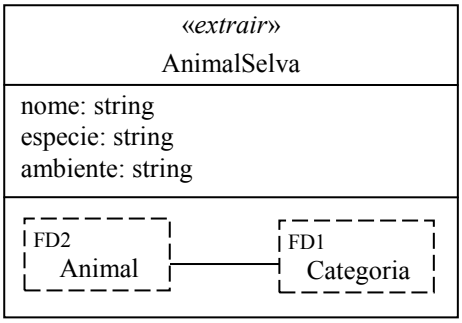


Figura 5.8 Modelar os predicados das relações unárias

[85] Nesta dissertação consideram-se apenas os *tipos de dados básicos* da UML 2.0, concretamente, os *tipos de dados real*, *integer*, *boolean* e *string*.

O designer pode utilizar as *expressões* de OCL para modelar os valores dos *atributos* utilizados para representar os predicados. As referidas *expressões* devem ser declaradas no contexto dos *objectos* das *classes* «*extrair*» e o corpo destas *expressões* deve ser respeitado por todos os *objectos* destas *classes* e em todos os momentos (*invariant*). As *expressões* de OCL mais usuais na referida modelação compreendem:

1. Os argumentos - *atributos* utilizados para representar os predicados ou constantes (valores reais, inteiros, booleanos e cadeias de caracteres).
2. As operações - operações de igualdade ou desigualdade que são independentes dos domínios de valores dos argumentos [operações = ou < >] e operações especializadas que são específicas dos domínios de valores dos argumentos: (1) reais ou inteiros, incluindo as operações aritméticas [operações *, +, -, /, etc.] e de comparação [operações >, <, >=, <=]; (2) booleanos [operações *and*, *or*, *xor*, *not*, *implies*, *if-then-else*, etc.]; (3) cadeias de caracteres [operações *toUpper*, *size*, *concat*, *substring*, etc.] (ver OMG 2003: (6) 1 - 13).

No exemplo corrente, utiliza-se a *expressão* expAnimalSelva no contexto dos *objectos* da *classe* «*extrair*» AnimalSelva para modelar os valores dos *atributos* nome, especie e ambiente utilizados para representar os predicados (ver Figura 5.9).

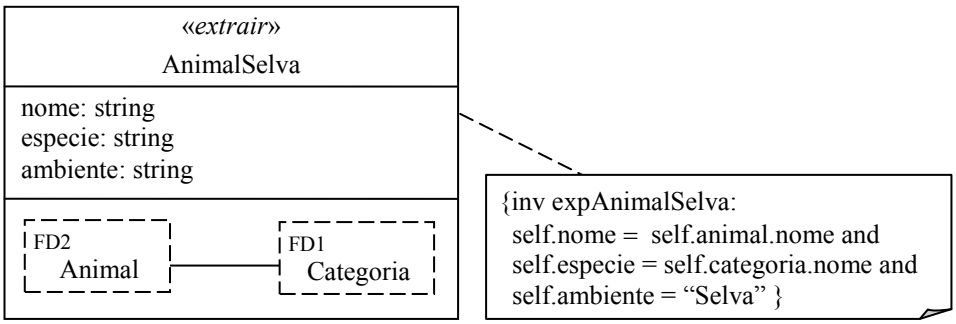


Figura 5.9 Modelar os valores dos predicados

Na *expressão* expAnimalSelva, utiliza-se a palavra reservada *self* para referir um *objecto* da *classe* AnimalSelva e os termos *self*.animal e *self*.categoria para referir os *objectos* descritos pelas *propriedades* dos *tipos* Animal e Categoria, respectivamente. A referida *expressão* tem como argumentos o *atributo* "nome" da coisa elementar do tipo "Animal", o *atributo* "nome" da coisa elementar do tipo "Categoria" e a constante "Selva", compreendendo ainda as operações de igualdade [=] (ver Figura 5.9).

O designer pode optar por omitir as *expressões* de OCL dos *diagramas*, declarando o contexto explicitamente nas *expressões*. No exemplo corrente, omite-se a *expressão* expAnimalSelva do *diagrama* especificando o contexto do seguinte modo:

{Context ME::AnimalSelva inv expAnimalSelva:
self.nome = self.animal.nome and
self.especie = self.categoria.nome and
self.ambiente = "Selva" }

5.3.1.3 Modelar os Predicados das Relações Binárias

O designer deve utilizar os *conectores* para modelar os predicados das relações binárias e o *tipo* dos *conectores* para indicar as *associações* que descrevem os referidos predicados nos modelos das fontes de dados (ver Definição 5.3 no ponto 5.3.1.1).

No exemplo corrente, utiliza-se o *conector* do *tipo* Recolhe (do FD2) para descrever os predicados "recolhe" da relação binária "o investigador recolhe medições" (ver Figura 5.10).

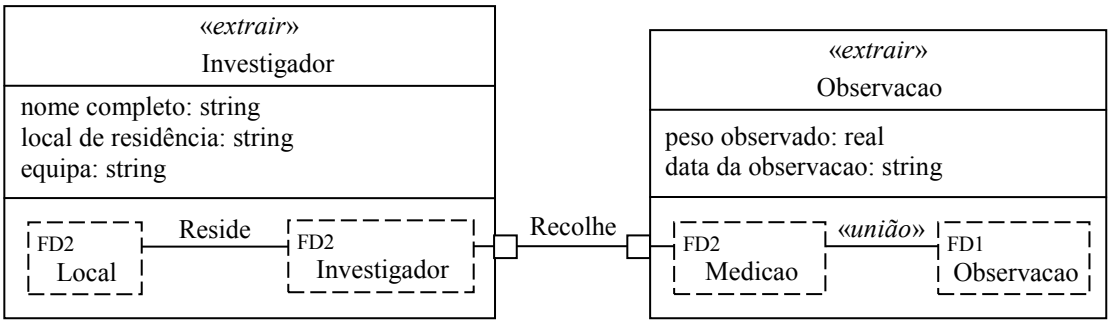


Figura 5.10 Modelar os predicados das relações binárias

Note-se que as coisas elementares que participam nas referidas relações são declaradas em diferentes *classes* «*extrair*». O designer deve utilizar os *portos* das *propriedades* para indicar as coisas predicadas que participam nas relações binárias. Os *portos* assumem os *tipos* das *propriedades* que representam as respectivas coisas predicadas.

No exemplo corrente, utilizam-se os *portos* dos *tipos* Investigador (do FD2) e Medicao (do FD2) para indicar que as coisas elementares dos tipos "Investigador" e "Medicao" participam nas relações que contêm o predicado "Recolhe", respectivamente (ver Figura 5.10).

5.3.2 Modelar as Regras

Neste passo, o designer deve modelar as regras que regulam os tipos de coisas relevantes que se extraem das fontes, concretamente, as regras relativas à cardinalidade dos predicados e as regras relativas ao valor das características das coisas elementares, bem como as regras que regulam o modo como as coisas desses tipos são extraídas.

5.3.2.1 Modelar as Regras Relativas à Cardinalidade dos Predicados

O designer deve utilizar as *multiplicidades* para modelar as regras relativas à cardinalidade dos predicados.

> **Definição 5.8** ***Multiplicidade*** - a *multiplicidade* de um dado elemento (*atributo* ou *extremo* do *conector*) é um intervalo de inteiros positivos que delimita as cardinalidades permitidas na instanciação desse elemento. São permitidos intervalos com limite superior infinito (representado por um *asterisco* *).

O designer deve utilizar: (1) as *multiplicidades* dos *atributos* das *classes* «*extrair*» para modelar as regras relativas à cardinalidade dos predicados das relações unárias; (2) as *multiplicidades* dos *extremos* dos *conectores* para modelar as regras relativas à cardinalidade dos predicados das relações binárias.

No exemplo corrente, utiliza-se as *multiplicidades* dos *atributos* da *classe* «*extrair*» Investigador para expressar que as coisas predicadas do tipo "Investigador" são caracterizadas por um nome completo e uma equipa - *multiplicidades* 1..1 - e, opcionalmente, por um local de residência - *multiplicidade* 0..1 (ver Figura 5.11).

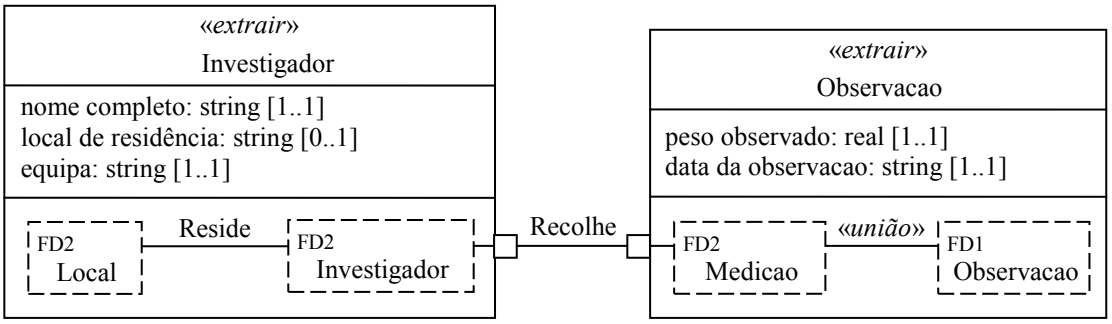


Figura 5.11 Modelar as regras relativas à cardinalidade nas relações unárias

No exemplo corrente, utilizam-se as *multiplicidades* dos *extremos* do *conector* do *tipo* Reside para modelar as regras relativas à cardinalidade dos predicados da relação binária composta pelos investigadores que residem num dado local.

Concretamente, utiliza-se a *multiplicidade* 1..* para expressar que um dado local tem um ou mais residentes e a *multiplicidade* 0..1 para expressar que um dado investigador pode residir num dado local. Nos *extremos* do *conector* com o *tipo* Recolhe, utiliza-se a *multiplicidade* 1..* para expressar que um dado investigador recolhe uma ou mais medições e a *multiplicidade* 1..1 para expressar que uma dada observação é recolhida por um investigador (ver Figura 5.12).

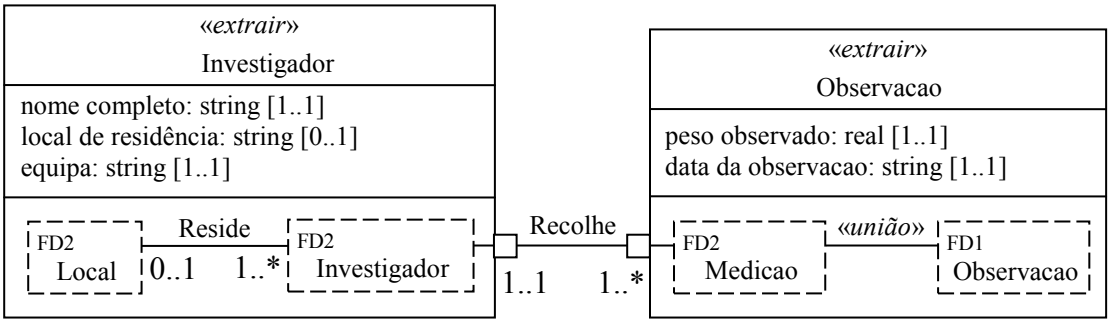


Figura 5.12 Modelar as regras relativas à cardinalidade nas relações binárias

5.3.2.2 Modelar as Regras Relativas ao Valor das Características

O designer deve utilizar as *expressões* de OCL para modelar as regras relativas aos valores das características das coisas elementares.

Deve declarar as *expressões* de OCL no contexto das *propriedades* para modelar as regras que devem ser cumpridas por todas as coisas elementares de um dado tipo e em todos os momentos (*invariant*).

No exemplo corrente, utiliza-se a *expressão* expIdadeEstudante no contexto da *propriedade* do *tipo* Estudante (do FD1) para expressar que os "Estudantes" são caracterizados por uma idade igual ou superior a 18 anos (ver Figura 5.13).

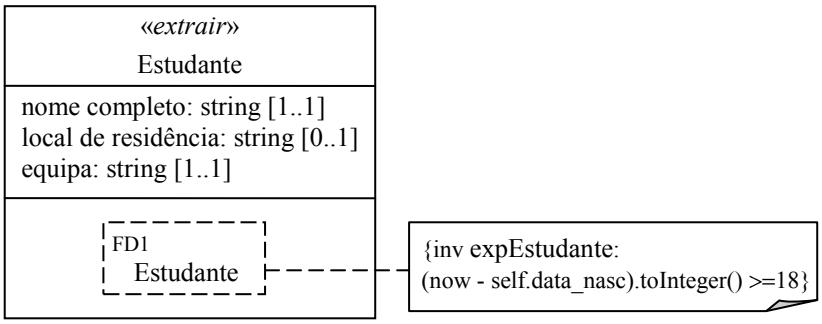


Figura 5.13 Modelar as regras cumpridas pelas coisas elementares

Por outro lado, o designer deve declarar as *expressões* de OCL no contexto dos *conectores* para modelar as regras cumpridas pelas coisas elementares que participam nas relações binárias de um dado tipo e em todos os momentos (*invariant*).

No exemplo corrente, utiliza-se a *expressão* expRegraAnimalSelva no contexto do *conector* da *classe* «*extrair*» AnimalSelva para expressar que, na relação binária do tipo "Animal da Selva", os animais classificados numa dada categoria taxionómica têm uma referência da espécie idêntica à referência da respectiva categoria taxionómica (ref_especie = ref_categoria). Note-se que os animais podem não participar na referida relação binária - *multiplicidade* 0..1 do *extremo* do *conector* (ver Figura 5.14).

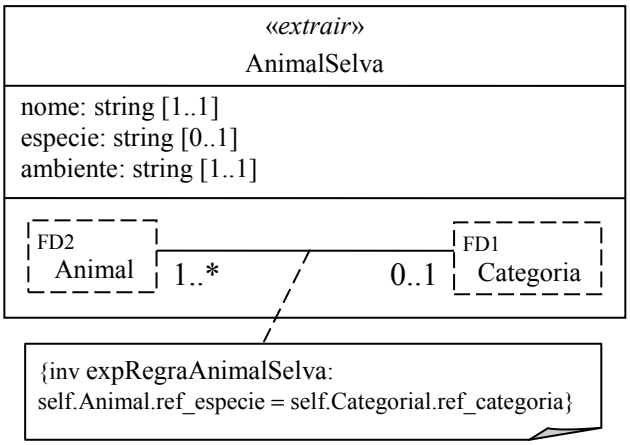


Figura 5.14 Modelar as regras cumpridas pelas coisas elementares das relações

O designer deve recorrer à *navegação* para modelar as regras que compreendem as características de múltiplas coisas elementares. Partindo de uma *propriedade* que representa uma dada coisa elementar, o designer pode *navegar* através dos *conectores* para utilizar as *propriedades* que representam outras coisas elementares.

No exemplo corrente, utiliza-se a *expressão* expRegraAnimalParque no contexto da *propriedade* do *tipo* Animal (do FD1) para expressar que o nome das espécies dos animais do parque é idêntico ao nome das categorias taxionómicas que classificam os referidos animais. Utiliza-se uma *navegação* desde a *propriedade* que representa o "Animal" - termo *self* - até à *propriedade* que representa as "Categorias taxionómicas" - termo *self*.Categoria (ver Figura 5.15).

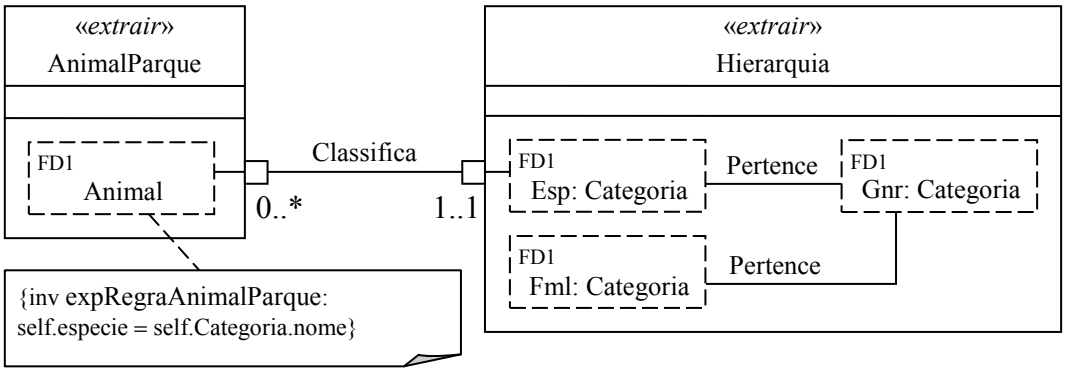


Figura 5.15 Modelar as regras que compreendem múltiplas coisas elementares

O designer deve especificar as *navegações*, atendendo à *multiplicidade* do *extremo* oposto desses *conectores*. O resultado de uma dada *navegação* pode ser: um único *objecto*, quando a referida *multiplicidade* é 1 ou 0..1; uma *colecção de objectos* (*Set*, *Bag* ou *Sequence*), quando a referida *multiplicidade* é n..m (m>n e m>1). As *navegações* que resultam em *colecções de objectos* envolvem a utilização de operações específicas [*select*, *reject*, *forAll*, *exists*, etc.].[86]

No exemplo corrente, a *navegação* da *expressão* expRegraAnimalParque resulta num único *objecto* da *classe* Categoria (do FD1), atendendo à *multiplicidade* 1..1 do *extremo* oposto do *conector* do tipo Classifica (ver Figura 5.15).

5.3.2.3 Modelar as Regras Relativas ao Modo como as Coisas são Extraídas

O designer deve atender às técnicas de extracção utilizadas para capturar os dados das fontes, distinguindo: (1) as *classes* «*extrair*» que descrevem apenas as coisas predicadas cujos predicados foram alterados entre dois períodos sucessivos de extracção (quando se utiliza uma técnica de extracção incremental); (2) as *classes* «*extrair*» que descrevem a totalidade das populações de um dado tipo (quando se utiliza uma técnica de extracção total).

O designer deve utilizar o *estereótipo* «*alterado*» para modelar as coisas predicadas cujos predicados foram alterados entre dois períodos sucessivos de extracção e omitir o referido *estereótipo* para modelar a totalidade da população de um dado tipo.

Def. 9 Estereótipo «*alterado*» - o *estereótipo* «*alterado*», quando aplicado sobre uma *classe*, indica que a referida *classe* descreve um conjunto de coisas

[86] Ver a descrição das operações de *colecções de objectos* de OMG (2003: 6.13-28).

predicadas cujos predicados foram alterados entre dois períodos sucessivos de extracção.

No exemplo corrente, utiliza-se o *estereótipo* «*alterado*» para indicar que a *classe* «*extrair*» Investigador descreve as coisas predicadas cujos predicados foram alterados entre dois períodos sucessivos de extracção, atendendo ao facto que se recorre a uma técnica de extracção incremental para extrair os dados que representam os "Investigadores". A omissão deste *estereótipo* na *classe* «*extrair*» Observacao, por sua vez, atende à utilização de uma técnica de extracção total para extrair os dados que representam as "Observações" (ver Figura 5.16).

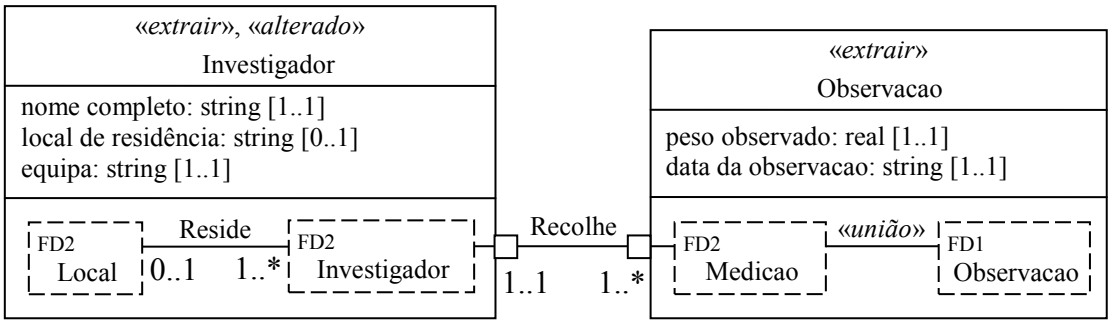


Figura 5.16 Modelar as regras relativas ao modo como as coisas relevantes são extraídas

6 CONCLUSÕES E FUTURAS DIRECÇÕES

Na introdução, definiu-se como objectivos principais tornar explícito e documentar o problema do refrescamento e apresentar um guião de modelação conceptual da tarefa de extracção de dados que possa enriquecer as fases subsequentes do desenho para a especificação formal do processo de refrescamento dos RDW. Neste capítulo, discute-se de que modo é que os resultados atingiram os objectivos definidos, sumariando-se os principais contributos desta dissertação e avaliando-se os limites do estudo (secção 5.1), e identificam-se algumas orientações para futuros trabalho (secção 5.2).

6.1 Contributos e Limites

Os resultados apresentados ao longo desta dissertação cumprem com os objectivos definidos e providenciam contributos importantes. Primeiro, faz-se a caracterização do problema do refrescamento dos RDW e mostra-se a sua complexidade e relevância. Segundo, organiza-se de forma compreensiva o domínio da extracção de dados concebido como um modelo intencional que é representado em UML, propondo um guião para apoiar o designer na modelação conceptual da tarefa de extracção de dados.

A caracterização do problema do refrescamento faz-se a dois níveis de análise.

No nível mais geral de análise, é discutido o conjunto de conceitos básicos e questões centrais dos SDW que caracterizam o contexto geral do problema do refrescamento, designadamente, as vantagens dos SDW no apoio à decisão face aos sistemas operacionais de OLTP, o problema de combinar os dados extraídos de fontes diferentes e apresentá-los num formato uniforme e global no RDW alvo e os vários componentes de arquitectura, incluindo as ferramentas de aquisição de dados das fontes, de armazenamento e gestão de dados *warehouse*, de acesso aos dados *warehouse* e a gestão dos metadados que suportam a funcionalidade do sistema.

Da discussão desses conceitos e questões decorrem duas características fundamentais dos SDW que permitem identificar a complexidade e relevância do problema do refrescamento.

A primeira característica diz respeito à separação entre, por um lado, as fontes de dados que gerem, de forma autónoma, dados operacionais correntes, detalhados e

atómicos que são acedidos através de aplicações de OLTP e, por outro, o *data warehouse* alvo que disponibiliza dados analíticos, históricos, sumariados (e detalhados) e integrados que suportam as aplicações de OLAP.

Enquanto que a qualidade dos dados, os esquemas e semânticas e os ritmos e proporções de alterações variam nas fontes de dados, o *data warehouse* assenta num esquema uniforme, global e consolidado de dados que são de acesso apenas para leitura, mas que têm que ser, periodicamente, refrescados, de modo a assegurar que reflectem as alterações que ocorrem nos dados das fontes que lhes servem de base, providenciando, deste modo, informação com qualidade para o apoio à decisão.

A segunda característica prende-se com o facto dos SDW, usualmente, gerirem um grande volume de dados distribuídos por uma hierarquia de camadas de dados com diferentes níveis de granularidade, desde os dados muito detalhados das fontes até aos de dados muito sumariados das bases de dados locais, como as estruturas de dados multidimensionais para o processamento OLAP ou os *data marts*, passando pelos dados detalhados dos ODS e pelos dados de baixa sumarização dos DW globais.

As alterações ocorridas nos dados das fontes devem ser capturadas, preparadas e propagadas ao longo dos RDW, dependendo dos constrangimentos impostos pelas propriedades de autonomia, distribuição e heterogeneidade das fontes de dados e pelo ambiente *warehouse* e dos requisitos dos utilizadores em termos da qualidade dos dados providenciados pelos diferentes RDW e, podendo as transacções que refrescam os dados entrar em conflito com as transacções que consultam os mesmos dados.

As principais características do problema do refrescamento são esclarecidas no segundo nível de análise, em dois passos. Primeiro, avaliando-se os constrangimentos e os requisitos dos utilizadores e descrevendo-se as tarefas de refrescamento. Segundo, analisando-se a literatura e discutindo-se as principais abordagens, as soluções propostas e questões em aberto, numa perspectiva crítica e comparada.

Este procedimento analítico permitiu caracterizar o refrescamento como um processo complexo que compreende várias tarefas técnicas e organizacionais que podem ser executadas de forma assíncrona, exigindo-se a sincronização durante a integração.

As diferenças entre o ambiente operacional e o ambiente *warehouse* devem ser supridas durante o processo de refrescamento, implicando que os dados extraídos de cada uma das fontes sejam limpos dos erros e inconsistências e sejam transformados num formato comum de dados e integrados no RDW alvo, tornando-se fundamental a reconciliação das diferenças estruturais e de semântica. Acresce o cálculo de dados

derivados para providenciar um nível comum de granularidade e construir as agregações, a preservação do histórico e o carregamento dos RDW.

Trata-se de um processo que envolve elevados custos e tempo de execução, exigindo várias técnicas de optimização, como as técnicas de extracção e de carregamento incremental, as técnicas de particionamento e de paralelismo e as técnicas de controlo de falhas.

A análise da literatura permitiu detectar duas abordagens diferentes sobre o problema do refrescamento que providenciam diferentes contributos para a solução do problema.

As abordagens da manutenção das vistas materializadas que reduzem o problema do refrescamento dos RDW à actualização das vistas materializadas e providenciam soluções assentes num conjunto de componentes de arquitectura, técnicas e algoritmos para suportar a manutenção incremental das vistas materializadas. As abordagens orientadas a modelos que entendem o refrescamento como um processo complexo que deve ser decomposto em diferentes tarefas e constrangimentos associados e que deve ser especificado logo na fase inicial do desenho de um SDW.

Esta dissertação inclui-se nas abordagens orientadas a modelos e providencia um guião de modelação conceptual que utiliza os conceitos propostos pelo FRISCO que permitem organizar um conjunto de aspectos que fazem parte do modelo intencional da extracção de dados representado em UML. Embora se trate de um modelo conceptual que analisa e descreve um dado domínio de extracção de dados, compreende aspectos organizados numa estrutura conceptual expressiva que se pode adequar a quaisquer cenários de refrescamento.

O guião proposto apoia o designer na modelação conceptual e oferece as construções que permitem representar os dados que se extraem das fontes, de acordo com as regras que permitem isolar e extrair os dados relevantes para a tomada de decisão, sem providenciar as soluções técnicas para a sua execução. Constitui-se, no entanto, como documentação essencial na fase inicial do desenho de um SDW que facilita a comunicação entre os *stakeholders* e que oferece metadados passivos para apoiar as subsequentes fases de desenho.

Ainda que se assumam os contributos divulgados na literatura, o modelo conceptual proposto distingue-se dos modelos providenciados por outros autores.

O metamodelo proposto pelo projecto SIRIUS é utilizado para modelar as tarefas de refrescamento asseguradas entre a área intermédia de armazenamento das

alterações que ocorrem nos dados das fontes e o DW global, mas remete as questões associadas à tarefa de extracção de dados para a utilização de monitores e de *wrappers*, apresentando uma aplicação prática desses componentes em cenários específicos de fontes de dados activas e fontes de dados replicadas.

Também relativamente às abordagens de modelação conceptual do domínio do processo de ETC adoptadas pelos investigadores do projecto Arktos e por Trujillo e Luján-Mora, os modelos propostos não modelam a tarefa de extracção, nem providenciam construções para representar aspectos do domínio estrito do refrescamento.

Por outro lado, os modelos conceptuais propostos são organizados em diferentes níveis de abstracção, sendo o atributo o nível mais detalhado, enquanto que o modelo que se apresenta situa-se no nível de detalhe da *classe*, uma vez que os *blueprints* da fase inicial do desenho de um SDW devem-se constituir como documentação de visualização rápida e compreensão fácil, de modo a melhorar a comunicação entre os *stakeholders*.

Os resultados apresentados nesta dissertação decorrem de um conjunto de opções metodológicas e de orientações de pesquisa que necessariamente impõem limites.

Desde logo pela inclusão de uns aspectos em detrimento de outros para tratar um dado problema. Por exemplo, os limites que resultam de uma dada orientação no sentido da modelação conceptual da tarefa de extracção de dados, muito ausente na investigação e muito reclamada na literatura e na prática, embora negligenciando-se os aspectos técnicos.

Por outro lado, a opção por representar os dados relevantes que se extraem das fontes, impõe alguns limites ao modelo proposto. O elevado nível de abstracção não permite incluir os aspectos associados à estrutura física dos componentes envolvidos, nomeadamente, das fontes de dados, dos *wrappers* e dos monitores que poderão ter um grande impacto na tarefa de extracção.

Também os aspectos dinâmicos da tarefa de extracção de dados, como o período e a frequência com se extraem os dados relevantes e como as mensagens transmitidas entre as fontes de dados e os componentes de extracção não são considerados. Admite-se que os diagramas da UML utilizados para capturar os aspectos estruturais (*diagramas de classes* e *diagramas de estruturas compostas*) possam ser complementados com os

diagramas de comportamento da UML para capturar os aspectos dinâmicos da tarefa de extracção de dados.

Do mesmo modo, as construções propostas foram utilizadas, em laboratório, na modelação de cenários específicos de extracção. Os cenários escolhidos são de pequena dimensão, envolvendo duas fontes de dados e um pequeno conjunto de tipos de coisas e de regras. Quando se utilizam as construções em cenários de maior dimensão e complexidade colocam-se as seguintes questões:

Como organizar grandes conjuntos de tipos de coisas relevantes de uma forma intuitiva? Em múltiplos *pacotes* de acordo com os conteúdos de informação? Tirando proveito dos mecanismos de encapsulamento da UML para permitir a navegação de um nível de maior detalhe para um nível de menor detalhe?

Como lidar com a complexidade das regras que regulam os tipos de coisas relevantes? Explorando as capacidades das *expressões* de OCL, das *multiplicidades* e dos *conectores* para modelar, com precisão, as regras mais complexas? Remetendo, para o modelo lógico, o detalhe das regras mais complexas?

6.2 Trabalhos Futuros

Um trabalho de investigação nunca está acabado. As questões tratadas suscitam outras que permitem prosseguir a pesquisa no domínio das abordagens orientadas a modelos, particularmente da tarefa de extracção de dados.

A aplicação do modelo conceptual proposto em cenários de refrescamento de grande dimensão é uma das questões que se continua a pesquisar. Concretamente, procuram-se soluções para facilitar a utilização das construções propostas em cenários que incluem uma grande variedade de fontes de dados e um grande número de tipos de coisas. Considera-se que os mecanismos de encapsulamento providenciados pela UML permitem organizar o modelo conceptual em vários níveis de detalhe e em grupos de tipos de coisas inter-relacionadas, facilitando a gestão e utilização de modelos de grande dimensão.

A validação do modelo conceptual proposto é outra linha de pesquisa que se prossegue. As construções da UML podem ser validadas, no domínio mais genérico, por um conjunto de restrições impostas pelo metamodelo da UML, mas as construções propostas por esta dissertação têm de ser validadas, no domínio mais específico, por um conjunto de restrições adicionais.

A validação do modelo conceptual proposto requer, por um lado, a identificação das restrições que devem ser cumpridas para assegurar que o modelo é correctamente especificado e, por outro, a utilização dos mecanismos de extensão da UML para expressar estas restrições no metamodelo. A identificação das referidas restrições é uma tarefa difícil, dada a grande variedade de domínios de aplicação dos SDW, as diferentes semânticas das construções propostas e as diferentes combinações de construções que podem ser utilizadas nos modelos conceptuais.

Finalmente, pretende-se especificar os mapeamentos entre as construções propostas para representar o modelo conceptual da extracção de dados e as construções das linguagens utilizadas para consultar e extrair os dados das fontes. Dada a grande variedade de fontes de dados e de métodos disponíveis para a extracção de dados, opta-se por focalizar as fontes de dados relacionais e as respectivas linguagens de consulta. Concretamente, procura-se identificar os procedimentos que permitem a transposição das construções do modelo conceptual proposto para um conjunto de vistas especificadas em SQL. Seguir-se-á a definição de um guião para apoiar o designer na transposição do modelo conceptual para o modelo lógico relacional. A concepção de um protótipo que permita automatizar alguns dos referidos procedimentos é também um objectivo de pesquisa futura.

7 BIBLIOGRAFIA

Aalst, W. (1997) Verification of workflow Nets, em: P. Azema e G. Balbo (eds.): *Application and Theory of Petri Nets*, Springer-Verlag: Berlin (*Lecture Notes in Computer Science*, 1248)

Aalst, W. e A. Hofstede (2002) Workflow patterns: On the expressive power of (Petri-net-based) workflow languages, em K. Jensen (ed.): *Proceedings of the 4th Workshop on the Practical Use of Coloured Petri Nets and CPN Tools*, Technical Report DAIMI PB-560: Aarhus,

Aalst, W. e A. Hofstede (2003) YAWL: Yet another workflow language (Revised version), *Technical Report*, FIT-TR-2003-04, Queensland University of Technology: Brisbane

Abelló, A. et al. (2001) A framework for the classification and description of multidimensional data models, em H. Mayr et al. (eds.): *Database and Expert Systems Applications. 12th International Conference*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 2113)

Adelberg, B. et al. (1997) The STRIP rule system for efficiently maintaining derived data, em J. Peckman et al. (eds.): *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Adelman, S. et al. (2003) *Impossible Data Warehouse Situations: Solutions from the Experts*, Addison Wesley: Boston

Agosta, L. (2002) Market overview update: ETL, *Technical Report RPA-032002-00021*, Giga Information Group

Agrawal, R. e R. Srikant (1994) Fast algorithms for mining association rules, em J. Bocca et al. (eds): *Proceedings of the 20th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Alter, S. (1980) *Decision Support Systems. Current Practice and Continuing Challenges*, Addison-Wesley: Boston

Amer-Yahia, S. e S. Cluet (2004) A declarative approach to optimize bulk loading into databases, *ACM Transactions on Database Systems*, 29(2): 233-81

Ananthakrishna, R. et al. (2002) Eliminating fuzzy duplicates in data warehouses, em *Proceedings of the 28th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Arnott, D. e G. Pervan 2005 A critical analysis of decision support systems research, *Journal of Information Technology*, 20: 67-87

Ashish, N. (2000) *Optimizing Information Mediators by Selectively Materializing Data,* Dissertação para obtenção do grau de DPhil em *Computer Science*, Faculty of the Graduate School, University of Southern California: California

Baekgaard, L. e L. Mark (1995) Incremental computation of time-varying query expressions, *IEEE Transactions on Knowledge and Data Engineering*, 7(4) 583-90

Baekgaard, L. e N. Roussopoulos (1997) Efficient refreshment of Data Warehouses views, *Technical Report: CS-TR-3642*, University of Maryland at College Park: College Park

Balsters, H. (2003a) A UML/OCL framework for designing mediated data federations, *New Economic Papers Research Report 03A17*, University of Groningen: Groningen

Balsters, H. (2003b) Object-oriented modeling of database federations, *New Economic Papers Research Report 03A18*, University of Groningen: Groningen

Barchman, R. et al. (1996) Mining business databases, *Communications of the ACM*, 39(11), 42-8

Batini, C. e M. Lenzerini (1984) A methodology for data schema integration in the entity relational model, *IEEE Transactions on Software Engineering*, 10(6), 650-64

Batini, C. et al. (1986) A comparative analysis of methodologies for database schema integration, *ACM Computing Surveys*, 18(4), 323-64

Baumöl, U. et al. (2000) Adapting the data warehouse concept for the management of decentralized heterogeneous corporations, *Journal of Data Warehousing*, 5(1): 35-43

Bercken, J. e B. Seeger (2001) An evaluation of generic bulk loading techniques, em P. Apers et al. (eds.): *Proceedings of the 28th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Bischoff, J. (1997) Introduction to data warehousing, em J. Bischoff e T. Alexander (eds.): *Data Warehouse: Practical Advice from the Experts*, Prentice-Hall: New Jersey

Blaha, M. e W. Premerlani (1998) *Object-Oriented Modeling and Design for Database Applications*, Prentice-Hall: Nova Iorque

Blakeley, J. et al. (1986) Updating derived relations: Detecting irrelevant and autonomously computable updates, em W. Chu et al. (eds.): *Proceedings of the 12th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Bock, C. (2004) UML 2 composition model, *Journal of Object Technology*, 3(10): 47-73

Booch, G. et al. (2005) *The Unified Modeling Language User Guide*, Addison-Wesley: Boston [1998]

Bouzeghoub, M. et al (1997) Designing data warehouse refreshment systems, *DWQ Technical Report*, DWQ Consortium

Bouzeghoub, M. et al (1998) A for developing efficient and customizable active rule systems, *DWQ Technical Report*, DWQ Consortium

Bouzeghoub, M. et al. (1999) Modeling data warehouse refreshment process as a workflow application, em S. Gatziu et al. (eds.): *Proceedings of the International Workshop on Design and Management of Data Warehouses*, CEUR-WS: Aachen

Bouzeghoub, M. et al. (2003) Data warehouse refreshment, em M. Jarke et al. (eds.): *Fundamentals of Data Warehouses*, Springer-Verlag: Berlim [1999]

Bouzeghoub, M. e V. Peralta (2004) A framework for analysis of data freshness, em F. Naumann e M. Scannapieco (eds.): *Proceedings of the International Workshop on Information Quality in Information Systems*, ACM Press: Nova Iorque

Brachman, R. et al. (1996) Mining business databases, *Communications of the ACM*, 39(11): 42-8

Bruckner, R. et al. (2002) Striving toward near real-time data integration for data warehouses, em Y. Kambayashi et al. (eds.) *Data Warehousing and Knowledge Discovery. 4th International Conference*, Springer-Verlag: Berlim (*Lecture Notes in Computer Science*, 2454)

Buzydlowsky, J. et al. (1998) A framework for object-oriented on-line analytical processing, em *Proceedings of the 1st ACM International Workshop on Data Warehousing and OLAP*, ACM Press: Nova Iorque

Cai, Y. et al. (2005) Optimized data loading for a multi-terabyte sky survey repository, em *Proceedings of the ACM/IEEE Conference on Supercomputing* IEEE Computer Society: Washington

Cali, A. et al. (2001) Models for information integration: turning local-as-view into global-as-view, em *Proceedings of the International 10th Workshop on Foundation of Models for Information Integration*, Springer-Verlag: Berlim

Cali, A. et al. (2003) Source integration for data warehousing, em M. Rafanelli (ed.): *Multidimensional Databases: Problems and Solutions*, Idea Group Publishing: Hershey

Calvanese, D. et al. (1998a) Information integration: Conceptual modeling and reasoning support, em *Proceedings of the 6th International Conference on Cooperative Information Systems*, IEEE Computer Society: Los Alamitos

Calvanese, D. et al. (1998b) Source integration in data warehousing, em *Proceedings of the 9th International Workshop on Database and Expert Systems Applications*, IEEE Computer Society: Los Alamitos

Calvanese, D. et al. (1999) A principled approach to data integration and reconciliation in data warehousing, em S. Gatziu et al. (eds.): *Proceedings of the International Workshop on Design and Management of Data Warehouses*, CEUR-WS: Aachen

Calvanese, D. et al. (2001) Data integration in data warehousing, *International Journal of Cooperative Information Systems*, 10(3): 237-271

Calvanese, D. et al. (2003) Source integration, em M. Jarke, M. et al. (eds.), *Fundamentals of Data Warehouses*, Springer-Verlag: Berlim [1999]

Casati, F. et al. (1995) Conceptual modeling of workflows, em M. Papazoglou (ed.): *Object-Oriented and Entity-Relationship Modelling. 14th International Conference*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 1021)

Casati et al. (1996) Workflow evolution, em B. Thalheim (ed.): *Proceedings of the 15th ER International Conference on Conceptual Modeling*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 1157)

Catarci, T. e M. Lenzerini (1993) Representing and using interschmes in cooperative information systems, *Journal of Intelligent and Cooperative Information Systems*, 2(4): 375-98

Cattell, R. e D. Barry, (2000) *The Object Data Standard*: *ODMG 3.0*, Morgan Kaufmann: São Francisco

Ceri, S. e J. Widom (1991) Deriving production rules for incremental view maintenance, em G. Lohman et al. (eds.): *Proceedings of the 17th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Chawathe, S. et al. (1994) The TSIMMIS project: Integration of heterogeneous information sources, em *Proceedings of the 100th Anniversary Meeting of the Information*, Processing Society of Japan: Tóquio

Chaudhuri, S. e U. Dayal (1997) An overview of data warehousing and OLAP technology, *SIGMOD Record*, 26(1): 65-74

Chaudhuri, S. et al. (2001) Database technology for decision support systems, *IEEE Computer*, 34(12), 48-55

Chaudhuri, S. et al. (2003) Robust and efficient fuzzy match for online data cleaning, em *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Chen, Z. (2002) *Intelligent Data Warehousing. From Data Preparation to Data Mining*, CRC Press LLC: Flórida

Codd, E. et al. (1993) Providing OLAP to user-analysts: An IT mandate, *Technical Report*, RF Codd & Associates: São José

Cheung, D. et al. (2001) Towards the building of a dense-region based OLAP system, *Data and Knowledge Engineering*, 36(1): 1-27

Colby, L. et al. (1996) Algorithms for deferred view maintenance, em J. Widom (ed.): *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Colby, L. et al. (1997) Supporting multiple view maintenance policies, em J. Peckman et al. (eds): *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Colliat, G. (1996) OLAP, relational and multidimensional database systems, *SIGMOD Record*, 25(3): 64-9

Dayal, U. et al. (1994) Active database systems, em W. Kim (ed.): *Modern Database Systems: The Object Model, Interoperability, and Beyond*, Addison-Wesley: Boston

Demarest, M (1997) *The politics of data warehousing*, *DSSResources.COM, 07/23/2004* http://dssresources.com/papers/features/demarest/demarest07232004.html

Demuth, B. e H. Hussmann (1999) Using UM/OCL constraints for relational database design, em R. France e B. Rumpe (eds.): *The Unified Modeling Language. Beyond the Standard. 2nd International Conference*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 1723)

Deveaux, J.-P. (2004) An in-memory compression technique for OLAP databases, *Report*, Dalhousie University: Halifax

Dhar, V. e R. Stein (1997) *Seven Methods for Transforming Corporate Data into Business Intelligence*, Prentice-Hall: New Jersey

Ding, L. et al. (1999a) Enhancing Existing Incremental View Maintenance Algorithms Using the Multi-Relation Encapsulation Wrapper, *Technical Report WPI-CS-TR-99-23*, Worcester Polytechnic Institute: Worcester

Ding, L. et al. (1999b) The MRE wrapper approach: enabling incremental view maintenance of data warehouses defined on multi-relation information sources, em *Proceedings of the 2nd ACM international Workshop on Data Warehousing and OLAP*, ACM Press: Nova Iorque

Doan, A. e A. Halevy (2005) Semantic integration research in database community: A brief survey, *AI Magazine*, 26(1): 83-94

Drazdzel, M. e R. Flynn (2000) Decision support systems, em A. Kent: *Encyclopedia of Library and Information Science*, M. Dekker: Nova Iorque, 67(30)

Eavis, T. (2003) Parallel Relational OLAP, Dissertação para obtenção do grau de DPhil, Department of Computer Science, Dalhousie University: Halifax

Embury, S. et al. (2001) Adapting integrity enforcement techniques for data reconciliation, *Information Systems*, 26: 657-89

English, L. (2000) Plain English on data quality. Information quality management: The next frontier, *DM Review Magazine*
http://www.dmreview.com/article_sub.cfm?articleId=2073

Engström, H. (2002) *Selection of Maintenance Policies for a Data Warehousing Environment. A Cost Based Approach to Meeting Quality of Service Requirements*, Tese para obtenção do grau de DPhil em *Computer Science*, University of Exeter: Suécia

Engström, H. et al. (2000) A Holistic Approach to the Evaluation of Data Warehouse Maintenance Policies, *Technical Report*, University of Skövde: Skövde

Engström, H. et al. (2003) Maintenance policy selection in heterogeneous data warehouse environments: A heuristics-based approach, em *Proceedings of 5th ACM International Workshop on Data Warehousing and OLAP*, ACM Press: Nova Iorque

Erik, T. (1997) Music of the cubes, *Database Programing & Design*, 10(10): 66-9

Fabret, M. et al. (1997) State of the art: Data warehouse refreshment, *DWQ Technical Report*, DWQ Consortium

Falkenberg, E. et al. (1998) *A Framework of Information System Concepts*, IFIP WG 8.1, Task Group FRISCO (cortesia do autor)

Fan, H. (2005) - *Investigating a Heterogeneous Data Integration Approach for Data Warehousing*. Tese para obtenção do grau de DPhil, School of Computer Science & Information Systems: Londres

Fayyad, U. (1998) Mining databases: Towards algorithms for knowledge discovery, *Bulletin of the Technical Committee on Data Engineering*, 21(1): 39-48

Fayyad, U. et al. (1996) From Data Mining to Knowledge Discovery in Databases, *AI Magazine*, 17(3): 37-54

Fayyad, U. et al. (2003), Summary from the KDD-03 pannel, data mining: The next 10 years, *ACM SIGKDD Explorations Newsletter*, 5(2): 191-6

Fenk, R. et al. (2000) Bulk loading a data warehouse build upon a UB-Tree, em *Proceedings of the International Database Engineering and Applications Symposium*, IEEE Computer Society: Los Alamitos

Firestone, J. (1998) Architectural evolution in data warehousing and distributed knowledge management architecture, *Executive Information Systems* (*White Paper*, 11)

Folkert, N. et al. (2005) Optimizing refresh of a set of materialized views, em *Proceedings of the 31st International Conference on Very Large Data Bases*, Endowment: Trondheim

Franconi, E. et al (2003) Multidimensional data models and aggregation, em M. Jarke et al. (eds.): *Fundamentals of data warehouses*, Springer-Verlag: Berlim [1999]

Friedrich, J. (2005) Meta-data version and configuration management in multi-vendor environments, em *Proceedings of the ACM SIGMOD international Conference on Management of Data*, ACM Press: Nova Iorque

Galhardas, H. (2001) *Nettoyage de Données: Modèle, Langage Déclaratif, et Algotithmes*, Tese para obtenção do grau de Doutor em *Informatique*, Université de Versailles Saint-Quentin-en-Yvelines : Versalhes

Galhardas, H. et al. (2000a) AJAX: an extensible data cleaning tool, em W. Chen et al. (eds.): *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Galhardas, H. et al. (2000b) An extensible framework for data cleaning, em *Proceedings of the 16th International Conference on Data Engineering*, IEEE Computer Society: San Diego [1999, *INRIA Technical Report*, 3742]

Garcia-Molina, H. et al. (1998) Expiring data in a warehousing, em A. Gupta et al. (eds.): *Proceedings of the 24th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Gatziu, S. et al. (1998) SIRIUS: An approach for data warehouse refreshment. *Technical Report*, Department of Computer Science, University of Zurich: Zurique

Golfarelli, M. et al. (2004) Beyond data warehousing: What's next in business intelligence?, em *Proceedings of the 7th ACM international workshop on Data warehousing and OLAP*, ACM Press; Nova Iorque

Gopalkrishnan, V. et al. (1999) Star/snow-flake schema driven object-relational data warehouse design and querie processing strategies, em M. Mohania e A. Tjoa (eds.): *Data Warehousing and Knowledge Discovery. 1st International Conference*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 1676)

Gorla, N. (2003) Features to consider in a data warehousing system, *Communications of the ACM*, 46 (11): 111-5

Grifin, T. e L. Libkin (1995) Incremental maintenance of views with duplicates, em M. Carey e D. Schneider (eds.): *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Gupta, A. e A. Blakeley (1995) Using partial information to update materialized views, *Information Systems*, 20(8): 641-62

Gupta, A. e I. Mumick (1995) Maintenance of materialized views: Problems, techniques and applications, *Bulletin of the Technical Committee on Data Engineering*, 18(2): 3-16

Gupta, A. e I. Mumick (eds.) (1999) *Maintenance of Materialized Views: Techniques, Implementations and Applications*, The MIT Press: Cambridge

Gupta, A. e I. Mumick (1999a) Applications of materialized views, em A. Gupta e I. Mumick (eds.): *Maintenance of Materialized Views: Techniques, Implementations and Applications*, The MIT Press: Cambridge

Gupta, A. e I. Mumick (1999b) Challenges in supporting materialized views, em A. Gupta e I. Mumick (eds.): *Maintenance of Materialized Views: Techniques, Implementations and Applications*, The MIT Press: Cambridge

Gupta, A. e I. Mumick (1999c) Introduction to views, em A. Gupta e I. Mumick (eds.): *Maintenance of Materialized Views: Techniques, Implementations and Applications*, The MIT Press: Cambridge

Gupta, A. e I. Mumick (1999d) Maintenance Policies, em A. Gupta e I. Mumick (eds.): *Maintenance of Materialized Views: Techniques, Implementations and Applications*, The MIT Press: Cambridge

Gupta, A. et al. (1993) Maintaining views incrementally, em P. Buneman e S. Jajodia (eds.): *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Gupta, A. et al. (1996) Data integration using self-maintainable views, em P. Apers et al. (eds.): *Advances in Database Technology. Proceedings of the 5th International Conference on Extending Database Technology*, Springer-Verlag: Londres (*Lecture Notes in Computer Science*, 1057)

Gupta, A. et al. (2001) Adapting materialized views after redefinitions: Techniques and a performance study, *Information Systems,* 26(5): 323-62

Gupta, H. (1997) Selection of views to materialize in a data warehouse, em F. Afrati e P. Kolaitis (eds.): *Database Theory. 6th International Conference*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 1186)

Gupta, H. e D. Srivastava (1999) The data warehouse of newsgroups, em C. Beeri e P. Buneman (eds.): *Database Theory. 7th International Conference*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 1540)

Hammer, J. et al. (1995) The Stanford data warehousing project, *Bulletin of the Technical Committee on Data Engineering*, 18 (2): 40-7

Han, J. e M. Kamber (2001) *Data Mining: Concepts and Techniques*, Morgan Kaufmann: São Francisco

Hand, D. et al (2001) *Principles of data mining*, MIT Press: Cambridge

Hakinpour, F. e A. Geppert (2001) Resolving semantic heterogeneity in schema integration: An ontology based approach, em *Proceedings of the International Conference on Formal Ontology in Information Systems*, ACM Press: Nova Iorque

Hanson, E. (1987) A performance analysis of view materialization strategies, em U. Dayal (ed.): *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Herfert, M. (2001) Managing and measuring data quality in data warehousing, em *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics*, 55-65

Herfert, M. e C. Hermann (2002) Proactive data quality management for data warehouse systems. A metadata based data quality system, em L. Lakshmanan (ed.): *Proceedings of 4th International Workshop on Design and Managment of Data Warehouses*, CEUR-WS: Aachen

Hernàndez, M. e S. Stolfo (1995) The merge/purge problem for large databases, em M. Carey e D. Schneider (eds.): *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Hull, R. (1997) Managing semantic heterogeneity in databases: a theoretical perspective, em *Proceedings of the 16th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, ACM Press: Nova Iorque

Hull, R. e G. Zhou (1996) A framework for supporting data integration using the materialized and virtual approaches, em J. Widom (ed.): *Proceedings of the ACM SIGMOD International Conference on Management of Data,* ACM Press: Nova Iorque

Huyn, N. (1997) Multiple-view self-maintenance in data warehousing environments, em M. Jarke et al. (eds.): *Proceedings of the 23rd International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Imhoff, C. et al. (2001) *Building the Customer-Centric Enterprise, Data Warehousing Techniques for Supporting Customer Relationship Management*, John Wiley & Sons: Nova Iorque

Inmon W. (1996) *Building the Data Warehouse*, John Wiley & Sons: Nova Iorque [1992]

Inmon, W. (1997) The data warehouse budget: Special feature from January 1997, *DM Review Magazine* (*Special Feature*)
http://www.dmreview.com/master.cfm?NavID=55&EdID=1315

Inmon, W. et al. (2001) *Corporate Information Factory*, John Wiley & Sons: Nova Iorque

ISO (1995) *International Organization for Standardization*, Information Technology – Database Languages – SQL Part 3: Call-Level Interface (SQL/CLI). ISO/IEC 9075-3

Jarke, M. e C. Quix (2003) Data warehouse research: issues and projects, em M. Jarke et al. (eds.), *Fundamentals of Data Warehouses*, Springer-Verlag: Berlim [1999]

Jarke, M. e Y. Vassiliou (1997) Data warehouse quality: a review of the DWQ project, em D. Strong e B. Kahn (eds.): *2nd Conference on Information Quality*, MIT Sloan School of Management: Cambridge

Jarke, M. et al. (1999) Architecture and quality in data warehouses. An extended repository approach, *Information Systems*, 24(3): 229-53

Jarke, M. et al. (eds.) (2003) *Fundamentals of Data Warehouses*, Springer-Verlag: Berlim [1999]

Jürgens, M. e H. Lenz (1999) Tree based indexes vs. bitmap indexes: a performance study, S. Gatziu et al. (eds.): *Proceedings of the International Workshop on Design and Management of Data Warehouses*, CEUR-WS: Aachen

Kaser, O. e D. Lemire (2003) Attribute value reordering for efficient hybrid OLAP, em *Proceedings of 6th ACM International Workshop on Data Warehousing and OLAP*, ACM Press: Nova Iorque

Katchaounov, T. (2003) *Query Processing for Peer Mediator Database*, Dissertação para obtenção do grau de DPhil em *Computer Science*, Department of Informayion Technology, Uppsala University: Uppsala

Keen,P. (1991) *Shaping the Future. Business Design through Information Technology*, Harvard Business School Press: Boston

Kim, W. (2005) On Metadata Management Technology: Status and Issues, *Journal of Object Technology*, 4(2): 41-7

Kimball, R. et al. (1998) *The Data Warehouse Lifecycle Toolkit: Export Methods for Designing, Developing and Deploying Data Warehouses*, Wiley Publishing: Indianapolis

Kimball, R e Caserta, J. (2004) *The Data Warehouse ETL Toolkit. Pratical Techniques for Extracting, Cleaning, Conforming, and Delivering Data*, Wiley Publishing: Indianapolis

Koschel, A. e P. Lockemann (1998) Distributed events in active database systems. Letting the genie out of the bottle, *Data and Knowledge Engineering*, 25(1-2): 11-28

Kühn, E. (2003) The zero-delay Data Warehouse: mobilizing heterogeneous databases, em J. Freytag et al. (eds.): *Proceedings of the 29th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Labio, W. e H. Garcia-Molina (1996) Efficient snapshot differential algorithms for data warehousing, em T. Vijayaraman et al. (eds.): *Proceedings of the 22nd International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Labio, W. et al. (1997) The WHIPS prototype for data warehouse creation and maintenance, em J. Peckman et al. (eds.): *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Labio, W. et al. (1999) Shrinking the warehouse update window, em A. Delis et al. (eds.): *Proceedings ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Labio, W. et al. (2000) Efficient resumption of interrupted warehouse loads, em *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Laudon, K. e J. Laudon, (2000) *Management Information Systems* Organization and Technology in the Networked Enterprise, Prentice-Hall: Englewood Cliffs

Lee, M. et al. (2000) IntelliClean: A knowledge-based intelligent data cleaner, em *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM Press: Nova Iorque

Lenzerini, M. (2002), Data integration: A theoretical perspective, em *Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, ACM Press: Nova Iorque

Liu, B. et al. (2002) Batch data warehouse maintenance in dynamic environments, em *Proceedings of the 11th International Conference on Information and Knowledge Management*, ACM Press: Nova Iorque

Liu, B. et al. (2003) A framework for optimizing view maintenance plans over distributed data sources, *Computer Science Technical Report Series*, Worcester Polytechnic Institute: Worcester

Low, W. et al. (2001) A knowledge-based approach for duplicate elimination in data cleaning, *Information Systems. Special Issue on Data Extraction, Cleaning and Reconciliation*, 26 (8): 585-606

Luján-Mora, S. e J. Trujillo (2003) A comprehensive method for data warehouse design, em H. Lenz et al. (eds.): *Proceedings of the 5th International Workshop on Design and Management of Data Warehouses*, CEUR-WS: Aachen

Luján-Mora, S. e J. Trujillo (2004) A data warehouse engineering process, em T. Yakhno (ed.): *Advances in Information Systems. 3rd International Conference*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 3261)

Luján-Mora, S. et al. (2004) Data mapping diagrams for data warehouse design with UML, em P. Atzeni et al. (eds.): *Conceptual Modeling. 23rd ER International Conference*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 3288)

Madhavan, J. et al (2001) Generic schema matching with Cupid, em P. Apers et al. (eds.): *Proceedings of the 28th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Madhavan, J. et al (2005) Corpus-based schema matching, em *21st International Conference on Data Engineering*, IEEE Computer Society: Los Alamitos

Mallach, E. (1994) *Understanding Decision Support Systems and Expert Systems*, Richard D. Irwin: New Jersey

Mallach, E. (2000) *Decision Support and Data Warehouse Systems*, McGraw-Hill: New Jersey

Mannino, M. e Z. Walter (2004) A Framework for Data Warehouse Refresh Policies, *Technical Report CSIS-2004-001*, University of Colorado at Denver: Denver

Marakas, G. M. (1999) *Decision Support Systems in the 21st Century*, Prentice-Hall: New Jersey

March, S. e A. Heyner (2003) *Integrated Decision Support: A Data Warehousing Perspective*, Vanderbilt University e University of South Florida
http://mis.temple.edu/sigdss/icis03/proceedings/DSSWorkshop03-March.pdf

Martin, E. et al. (1994) *Managing Information Technology*, Macmilan Publications: Englewood Cliffs

Miliauskaitè, E. e N. Nemuraitè (2005) Representation of integrity constraints in conceptual models, *Information Technology and Control*, 34(4): 355-65

Monge, A. (2000) Matching algorithms within a duplicate detection system, *Bulletin of the Technical Committee on Data Engineering*, 23(4): 14-20

Monge, A. e C. Elkan (1997) An efficient domain-independent algorithm for detecting approximately duplicate database records, em *Proceedings of the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*

Moro, G. e C. Sartori (2001) Incremental maintenance of multi-source views, em *Proceedings of the 12th Australasian Database Conference*, IEEE Computer Society, Washington

Müller, H. e J-C Freytag (2003) Problems, methods, and challenges in comprehensive data cleansing, *Technical Report HUB-IB-164*, Humboldt University: Berlim

Müller, H. et al. (2004) Mining for patterns in contraditory data, em *Proceedings of the International Workshop on information Quality in information Systems*, ACM Press: Nova Iorque

Mumick, I. et al. (1999) Maintainance of data cubes and summary tables, em A. Gupta e I. Mumick (eds.): *Maintenance of Materialized Views: Techniques, Implementations and Applications*, The MIT Press: Cambridge

Navathe et al. (1984), Relationship merging in schema integration, em U. Dayal et al. (eds.): *Proceedings of the 10th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Nguyen, T e A. Tjoa (2003) Zero-latency data warehousing for heterogeneuos data sources and continuos dada streams, em G. Kotsis et al. (eds.): *The 5th International Conference on Information Integration and Web-Based Applications and Services*, Oesterreichische Computer Gesellschaft: Österreich

Nunes, M. e O'Neil, H. (2003) *Fundamental de UML*, Editora de Informática: Lisboa [2001]

Nutt, W (2003) Query processing and optimization, em M. Jarke et al. (eds.): *Fundamentals of Data Warehouses*, Springer-Verlag: Berlim

OMG (2003) Unified Modeling Language Specification v1.5, http://www.uml.org/#uml1.5

OMG (2001) *Common Warehouse Metamodel Specification* v1.0 http://www.omg.org/technology/cwm

OMG (2005) The Unified Modeling Language Superstructure v2.0 http://www.uml.org/#uml2.0

Ouksel, A. e A. Sheth (1999) Semantic interoperability in global information systems. A brief introduction to the research area and the special section, *SIGMOD record* 28(1): 5-12

Özsu, T. e P. Valduriez (1999) *Principles of Distributed Database Systems*, Prentice-Hall: New Jersey

Park, C. et al. (2002) Finding an efficient rewriting of OLAP queries using materialized views in data warehouses, *Decision Support Systems*, 32(4): 379-99

Pedersen, T. e C. Jensen (2001) Multidimensional database technology, *IEEE Computer*, 34(12), 40-6

Pelozo. C. (2003) *A Requirements Engineering Approach for Object-Oriented Conceptual Modeling*, Tese para obtenção do grau de DPhil em *Computer Science*, Information Systems and Computation Departement, Valencia Polytechnic University: Valencia: California

Piatetsky-Shapiro, G. (2000) Knowledge discovery in databases: Ten years after, *SIGKDD Explorations*, 1(2), 59-61

Poole, J. et al. (2003) *Common Warehouse Metamodel. Developer's Guide*, Wiley Publishing: Indianapolis

Power, J. (2000) *Supporting Business Decision-Making. Good Is Essential for Fact-Based decision-Making*, Edição do Autor http://dssresources.com/dssbook/ch1sbdm.pdf

Power, J. (2001) *Supporting Decision-Makers: An Expanded Framework*, Informing Science http://proceedings.informingscience.org/IS2001Proceedings/pdf/PowerEBKSupp.pdf

Quass, D. e J. Widom (1997) On-line warehouse view maintenance, em J. Peckman et al. (eds.): *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Quass, D. et al. (1996), Making views self-maintainable for data warehousing, *Proceedings of the 4th International Conference on Parallel and Distributed Information Systems*, IEEE Computer Society, Washington

Quix, C. (1999) Repository support for data warehouse evolution, em S. Gatziu et al. (eds.): *Proceedings of the International Workshop on Design and Management of Data Warehouses*, CEUR-WS: Aachen

Ram, P. e L. Do (2000) Extracting delta for incremental data warehouse maintenance, em *Proceedings of the 16th International Conference on Data Engineering*, IEEE Computer Society: Washington

Rahm, E. e H. Do (2000) Data cleaning. Problems and current approaches, *Bulletin of the Technical Committee on Data Engineering*, 23(4): 3-13

Rahm, E e P. Bernstein (2001) On matching schemas automatically, *Microsoft Research Technical Report*, MSR-TR-2001-17

Raman, V. e J. Hellerstein (2001) Potter's Whell. An interactive data cleaning system em P. Apers et al. (eds.): *Proceedings of the 28th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Ramakrishnan, R. e J. Gehrke (2003), *Database Management Systems*, McGraw Hill: Nova Iorque

Rizzi, S. e I.-Y. Song (2004) Report on the ACM 6th International Workshop on Data Warehousing and OLAP, *ACM SIGIR FORUM*, 38(1): 61-4

Robertson, S e J. Robertson (1999) *Mastering the Requirements Process*, Addison-Wesley: Boston

Rosana, L. et al. (2003) Performance tests in data warehousing ETLM process for detection of changes in data origin, em Kambayashi, Y. et al. (eds.): *Data Warehousing and Knowledge Discovery, 5th International Conference*, Springer-Verlag: Berlim

Roth, P. (1995) *Data Warehousing and Decision Support. The State of the Art*, Spiral Books: Manchester

Roussopoulos, N. (1998) Materialized Views and Data Warehouses *SIGMOD Record*, 27(1): 21-6

Roussopoulos, N. et al. (1995) The ADMS Project: Views 'R' Us, *Bulletin of the Technical Committee on Data Engineering*, 18(2): 29-39

Rundensteiner, E. (ed.) (1999) *Bulletin of the Technical Committee on Data Engineering*. *Special Issue on Data Transformations*, IEEE Computer Society, Los Alamos

Sheth, A. P. (1999) Changing focus on interoperability in information systems. From system, syntax, structure to semantics, em M. Goodchild, M. et al. (eds): *Interoperating Geographic Information Systems*, Kluwer Publishers: Boston

Sheth, A. P. e J. A. Larson (1990) Federated database systems for managing distributed, heterogeneous, and autonomous databases, *ACM Computing Surveys*, 22(3): 183-236

Silberschatz, A. et al. (2006) *Database System Concepts*, McGraw-Hill: Bóston [1986]

Simitsis, A. (2003) Modeling and managing ETL process, em: M. Scholl T. Grust (ed.): *Proceedings of the 29th International Conference on Very Large Data Bases*, CEUR-WS: Berlim

Simitsis, A (2005) Mapping conceptual to logical models for ETL processes, em Song, I.-Y. e J. Trujillo (eds.): *Proceedings of the 8th ACM International Workshop on Data Warehousing and OLAP*, ACM Press: Nova Iorque

Spaccapietra, S. et al. (1992) Model independent assertions for schema integration of heterogeneous schemas, *The VLDB Journal*, 1(1): 81-126

Staudt, M. et al. (1999a) Metadata management and data warehousing, *Technical Report*, 21, University of Zurich: Zurique

Staudt, M. et al. (1999b) The role of metadata for data warehousing, *Technical Report*, 99, University of Zurich: Zurique

Strange, K (2002) ETL was the key to his data warehouse success, *Technical Report CS-15-3143*, Gartner

Sung, S. et al. (2002) A fast filtering scheme for large database cleaning, em *Proceedings of the 11th International Conference on information and Knowledge Management*, ACM Press: Nova Iorque

Theodoratos, D. e M. Bouzeghoub (1999) Data currency quality factors in data warehouse design, em S. Gatziu et al. (eds.): *Proceedings of the International Workshop on Design and Management of Data Warehouses*, CEUR-WS: Aachen

Theodoratos, D. e M. Bouzeghoub (2001) Data concurrency quality factors in data warehouse design, *Journal of Cooperative Information Systems*, 10(3): 299-326

Theodoratos, D. e T. Sellis (1997) Data warehouse configuration, em M. Jarke et al. (eds.): *Proceedings of the 23rd International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Trujillo, J. e Luján-Mora, S. (2003) A UML based approach for modeling ETL processes in data warehouses, em I. Song et al. (eds.): *Proceedings of the 22nd ER International Conference on Conceptual Modeling*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 2813)

Turban, E. et al. (2001) *Information Technology for Strategic Advantage*, John Wiley & Sons: Nova Iorque

Ullman, J. D. (1997) Information integration using logical view, em F. Afrati e P. Kolaitis (eds.): *Proceedings of the 6th ICDT Internacional Conference on Database Theory*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 1186)

Vaduva, A e K. Dittrich (2001) Metadata management for data warehousing: Between vision and reality, em M. Adiba et al. (eds.): *Proceedings of the International Database Engineering*, IEEE Computer Society: Washington

Vaduva, A. et al. (2001) M4 - A metamodel for data preprocessing, *Proceedings of the 4th ACM International Workshop on Data Warehousing and OLAP*, ACM Press: Nova Iorque

Vassiliadis, P. (2000a) *Data Warehouse Modeling and Quality Issues*, Tese para obtenção do grau de PhD em *Computer Science*, Departament of Electrical and Computer Engineering, National Technical University of Athens: Atenas

Vassiliadis, P. (2000b) Gulliver in the land of data warehousing: practical experiences and observations of a researcher, em M. Jeusfeld et al. (eds.): *Proceedings of the 2nd International Workshop on Design and Management of Data Warehouses*, CEUR-WS: Aachen

Vassiliadis, P et al. (2000a) ARTKOS: A tool for data cleaning and transformation in data warehouse environments, *Bulletin of the Technical Committee on Data Engineering*, *Special Issue on Data Cleaning*, 23(4): 42-7

Vassiliadis, P. et al. (2000b) A model for data warehouse operational processes, em W. Benkt e L. Bergman (eds.): *Advanced Information Systems Engineering, 12th International Conference*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 1789)

Vassiliadis, P. et al. (2001a) ARKTOS: Towards the modeling, design, control and execution of ETL processes, *Information Systems*, 26(8): 537-61

Vassiliadis, P. et al. (2001b) Data warehouse process management, *Information Systems*, 26(3): 205-36

Vassiliadis, P. et al. (2002a), Conceptual modeling for ETL processes, em *Proceedings of the 5th ACM international Workshop on Data Warehousing and OLAP*, ACM Press: Nova Iorque

Vassiliadis, P. et al. (2002b), Modelling ETL activities as graphs, em L. Lakshmanan (ed.): *Proceedings of the 4th International Workshop on Design and Management of Data Warehouses*, CEUR-WS: Aachen

Vassiliadis, P et al. (2003) A framework for the design of ETL scenarios, em J. Eder e E. Missikoff (eds.): *Advanced Information Systems Engineering, 15th International Conference*, Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 2681)

Vassiliadis, P et al. (2005a) A generic and customizable framework for the design of ETL scenarios, *Information Systems* 30(7): 492-525

Vassiliadis, P et al (2005b) Blueprints and measures for ETL workflows, em: L. Delcambre et al. (eds.): *Conceptual Modeling. ER 24th International Conference*, Springer-Verlag: Berlim (*Lecture Notes in Computer Science*, 3716)

Vassiliadis, P et al (2005c) Blueprints for ETL workflows, (versão longa de Simitsis, A. et al. Graph-based modeling of ETL activities with multi-level transformations and updates), em: A. Tjoa e J. Trujillo (eds.): *Data Warehousing and Knowledge Discovery. 7th International Conference*, Springer-Verlag: Berlim.

Vassiliadis, P. e T. Sellis (1999) A survey on logical models for OLAP databases, *SIGMOD Record*, 28(4): 64-9

Vavouras, A. et al. (1999a) The SIRIUS approach for refreshing data warehouses incrementally, Buchmann, A. (ed.): *Datenbanksysteme*, Springer-Verlag: Berlim

Vavouras, A. et al. (1999b) Modeling and executing data warehouse refreshment process**,** *Proceedings of the International Symposium on Database Applications in Non-Traditional Environments*, IEEE Computer Society: Washington

Vavouras, A. et al. (2000) Modeling and executing data warehouse refreshment process**,** *Technical Reports*, Universität Zürich: Zurique

Vavouras, A. (2002) *A Meta-Driven Approach for Data Warehouse Refreshment*, dissertação para obtenção do grau de *Doktor* em *Informatik*, Universität Zürich: Zurique

Wang, X. et al. (2000) *A Performance Analysis of View Maintenance Techniques for Data Warehouses* http://www.cs.ou.edu/~database/documents/wgz00.pdf

Ward, J. et al. (1990) *Strategic Planning for Information Systems*, John Wiley & Sons: Nova Iorque

Widom, J. (1995) "Research Problems in Data Warehousing", em N. Pissinou et al. (eds.): *Proceedings of 4th International Conference on Information and Knowledge Management*, ACM Press: Nova Iorque

Wiederhold, G. e R. Elmasri (1979) The structural model for database design, em P. Chen (ed.): *Proceedings of the 1st International Conference on the Entity Relationship Approach To Systems Analysis and Design*, ACM Press: North Holland

Wiederhold, G. (1992a) Mediators in the architecture of future information systems, *IEEE computer* 25(3): 38-49

Wiederhold, G. (1992b) Intelligent integration of diverse information, em T. Finin et al. (eds.): *Information and Knowledge Management*. *Expanding the Definition of Database. 1st International Conference,* Springer-Verlag: Munique (*Lecture Notes in Computer Science*, 2681)

Wiener, J. e J. Naughton (1994) *Bulk loading into an OODB: a performance study*, em J. Bocca et al. (eds.): *Proceedings of the 20th International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Wiener, J. et al. (1996) A system prototype for warehouse view maintenance. *Technical Report*, Stanford University: Stanford

Winkler, W. (2003) Data cleaning methods, em *Proceedings of the 9th International Conference on Knowledge Discovery and Data Mining*, ACM Press: Washington

Winter, R. e M. Meyer (2001) Organization of data warehousing in large service companies. A matrix approach based on data ownership and competence centers, *Journal of Data Warehousing*, 6(4): 23-9

Winter, R. e B. Strauch (2004) Information requirements engineering for data warehouse systems, em *Proceedings of the 2004 ACM Symposium on Applied Computing*, ACM Press: Nova Iorque

Wu, M. e A. Buchmann, A. (1997), Research issues in data warehousing, em K. Dittrich e A. Geppert (eds.): *Datenbanksystem in Burö Technik und Wissenschaft. GI-Fachtagung*, Springer-Verlag: Munique (*Informatik Aktuell*)

X/Open (1995) *Data management. SQL Call Level Interface*, X/Open Company: Berkshire [1992]

Xu, M. e C. Ezeife (2000) Maintaining horizontally partitioned warehouse views, em Y. Kambayashi et al. (eds.): *Data Warehousing and Knowledge Discovery. 2nd International Conference*, Springer-Verlag: Berlim (*Lecture Notes in Computer Science*, 1874)

Yan, L. et al. (2001) Data-Driven Understanding and Refinement of Schema Mappings, *ACM SIGMOD*, USA: California,

Yang J. et al. (1997) Algorithms for Materialized View Design in Data Warehousing Environment, Jarke, M. et al. (eds.): *Proceedings of the 23rd International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Yang, J. (2001) Temporal Data Warehousing, Tese para obtenção do grau de PhD em *Computer Science*, Stanford University: Stanford

Yang, J. e J. Widom (1998) Maintaining temporal views over non-temporal information sources for data warehousing, em H. Schek et al. (eds.): *Advances in Database Technology. Proceedings of the 6th International Conference on Extending Database Technology*, Springer-Verlag: Berlim (*Lecture Notes in Computer Science*, 1377)

Yang, J. et al. (1997) Framework for Designing Materialized Views in Data Warehousing Environment, em *Proceedings of the 23rd International Conference on Very Large Data Bases*, Morgan Kaufmann: São Francisco

Zhou, G. et al. (1995) Data integration and warehousing using H20, *Bulletin of the Technical Committee on Data Engineering*, 18(2): 29-40

Zhuge, Y et al. (1995) View maintenance in a warehousing environment, *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM Press: Nova Iorque

Zhuge, Y et al. (1996) The strobe algorithms for multi-source warehouse consistency, P*roceedings of the 4th International Conference on Parallel and Distributed Information Systems*, IEEE Computer Society: Washington

Zhuge, Y et al. (1997) Multiple view consistency for data warehousing, *Proceedings of the 13th International Conference on Data Engineering*, IEEE Computer Society: Washington

Zhuge, Y et al. (1998) Consistency algorithms for multi-source warehouse view maintenance, *Journal of Distributed and Parallel Databases*, 6(1): 7-40

INTERNET

Arktos http://www.dblab.ntua.gr/

Carnegie Mellon Sofware Engineering Institute (2004), What is a CASE Environment? *Computer-Aided Software Engineering (CASE) Environments*
http://www.sei.cmu.edu/legacy/case/case_whatis.html

Data Cleaning Project
http://www.research.microsoft.com/dmx/datacleaning/default.aspx

Database Group, University of Toronto http://www.cs.toronto.edu/db

Foundations of Data Warehouse Quality (DWQ) http://www.dbnet.ece.ntua.gr/~dwq/

ORACLE (2002), Oracle Call Interface Programmer's Guide, Release 2 (9.2), Oracle Corporation
http://www.stanford.edu/Dept/itss/docs/oracle/9i/appdev.920/a96584/toc.html

Supporting Incremental Refreshment of Information WarehoUseS (SIRIUS)
http://www.ifi.unizh.ch/dbtg/projects/sirius/sirius.html

WareHouse Information Prototype Stanford (WHIPS)
http://www-dbstanford.edu/warehousing/warehouse.html

WorkFlow Management Coalition (WfMC) http://www.wfmc.org

The Stanford-IBM Manager of Multiple Information sources (TSIMMIS),
http://www-db.stanford.edu/tsimmis/tsimmis.html

8 APENDICES

8.1 P Vassiliadis et al.

8.1.1 Notações Gráficas e Conceitos

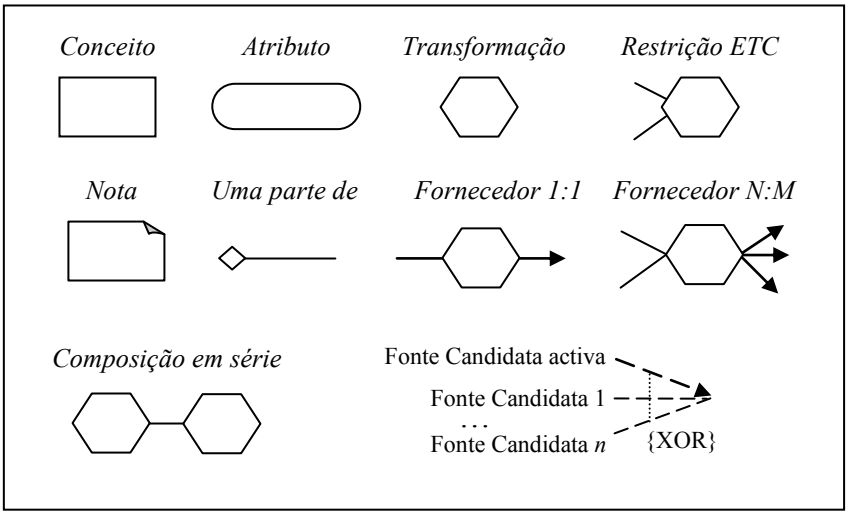


Figura 8.1 Notações gráficas

Para representar os dados das fontes e do DW global, propõe-se a utilização (1) do *conceito*, enquanto construção abstracta que representa uma entidade na fonte de dados ou no DW global; (2) do *atributo*, entendido como uma propriedade de uma entidade que pode assumir um conjunto de valores; (3) da relação do tipo *uma parte de*, que estabelece a associação entre o *atributo* e o respectivo *conceito*, para enfatizar o facto de um *conceito* ser descrito por um conjunto de *atributos* (Vassilidadis et al. 2002a).

As relações entre os *conceitos* são representadas pelas relações dos tipos *candidato* e *candidato activo* e as relações do tipo *fornecedor*. A relação entre *conceitos* é entendida como um mapeamento entre um ou mais *conceitos* de *input* e um ou mais *conceitos* de *output*, sugerindo-se a utilização de uma construção que represente o fluxo de dados na direcção do *input* para o *output*. A relação do tipo *candidato* permite capturar a existência de mais do que uma entidade fonte candidata (por exemplo, tabela, ficheiro, etc.) a popular uma entidade no DW alvo. Para representar a escolha de uma entidade para popular uma entidade alvo, quando existe um conjunto de entidades candidatas alternativas, sugere-se a utilização de uma relação do tipo *candidato activo*.

Transformações são definidas como abstracções que representam a parte ou o módulo completo de código de programação que executa uma tarefa do processo de ETC, distinguindo-se: (1) as operações de limpeza e filtragem de dados, como a verificação de integridade referencial ou a detecção de valores nulos; (2) as operações de transformação dos dados das fontes no formato comum dos dados do DW.

A *transformação* é representada formalmente sobre uma relação entre *conceitos* ou *atributos* (relação do tipo *fornecedor*) e compreende os conjuntos de *atributos* de *input* e de *output* e a representação da natureza e da semântica da transformação.

Para estender a representação da natureza e da semântica da *transformação*, sugere-se a utilização de uma *nota*, onde é possível especificar as operações concretizadas sobre os dados recorrendo, por exemplo, à linguagem natural ou a uma linguagem de programação.

As *restrições de ETL* são definidas como abstracções que permitem capturar a existência de uma imposição sobre as instâncias de um *conceito* que as obriga a respeitar uma condição ou requisito.

É ainda apresentada uma *palete* com os símbolos das *transformações* que o designer utiliza com maior frequência:

1 A aplicação de um *filtro* sobre um *atributo* de um *conceito* que o obriga a respeitar uma determinada condição, por exemplo, assumir um valor único, não assumir um valor nulo ou assumir um valor pertencer ao domínio dos números inteiros
2 A *transformação unária* efectuada sobre um *conceito*, como a sua projecção ou agregação
3 A *transformação binária* efectuada sobre dois ou mais *conceitos*, como a união, a diferença ou a detecção de alterações entre dois *conceitos*
4 A *composição de transformações*, como a definição do tipo de carregamento de uma dimensão (*slowly changing dimensions*) ou a conversão do tipo de dados
5 A *operação de transferência* de um *ficheiro* como a transferência de um ficheiro de uma fonte de dados para a DSA por *File Transfer Protocol* ou a compressão/descompressão de um ficheiro
6 A *operação efectuada sobre um ficheiro*, como a conversão EBCDIC para ASCII ou a ordenação de um ficheiro.

8.2 S Luján-Mora e J Trujillo

8.2.1 *ETL Profile*

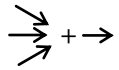
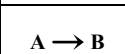
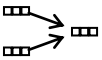
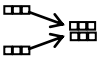
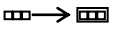
Mecanismo de ETL	Descrição	Ícone
«*Agregation*»	Agrega (soma, média, valor máximo/mínimo, contagem, etc.) os dados com base num dado critério	
«*Conversion*»	Altera o tipo de dados e formata ou deriva novos dados (atributos derivados) a partir dos dados existentes	
«*Filter*»	Filtra e descarta dados indesejados, e verifica a qualidade dos dados através de restrições	
«*Incorrect*»	Reencaminha os dados que se encontrem incorrectos ou que tenham sido descartados para um alvo diferente para verificação posterior, que pode apenas ser utilizado com o «*Filter*», o «*Loader*» e o «*Wrapper*»	
«*Join*»	Junta 2 fontes de dados relacionadas entre elas por um ou mais atributos	
«*Loader*»	Carrega dados num alvo de um processo de ETL (numa tabelas de factos ou dimensão de um DW)	
«*Log*»	Guarda a actividade de um mecanismo de ETL de forma a auditar o processo, assistir na limpeza de dados, no *tuning* da performance, etc.	
«*Merge*»	Integra 2 ou mais fontes de dados com atributos compatíveis	
«*Surrogate*»	Cria chaves únicas para o DW (*surrogate keys*), que são utilizadas para substituir as chaves originais das fonte	
«*Wrapper*»	Transforma uma fonte de dados nativa numa fonte de dados baseada em registos (*record based data source*)	

Tabela 8.1 Mecanismos de ETL e respectivos ícones

8.2.2 *Data Mapping Profile*

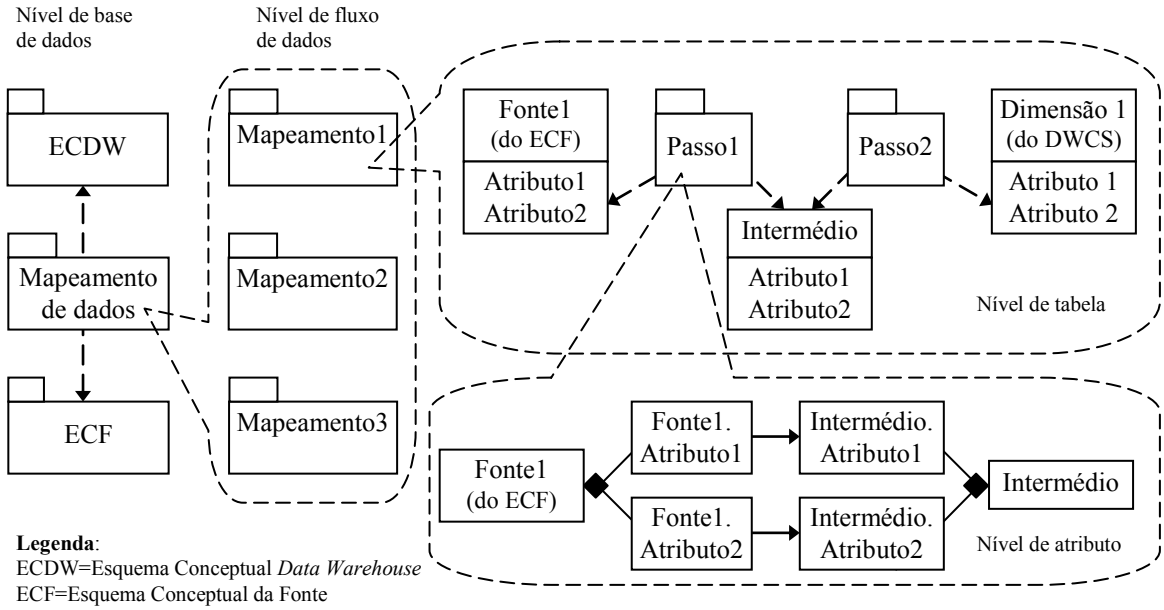


Figura 8.2 Mapeamentos nos diferentes níveis de detalhe