Universidade do Minho

Escola de Engenharia

Departamento de Informática

Sistemas de *Data Webhousing*: Análise, Desenho, Implementação e Exploração de Sistemas Reais

Eurico Alexandre Teixeira Borges

Dissertação de Mestrado

2004

Sistemas de *Data Webhousing*: Análise, Desenho, Implementação e Exploração de Sistemas Reais

Eurico Alexandre Teixeira Borges

Dissertação apresentada à Universidade do Minho para obtenção do grau de Mestre em Informática, na especialidade de Sistemas Distribuídos, Comunicações por Computador e Arquitectura de Computadores, elaborada sob orientação do Professor Doutor Orlando Manuel de Oliveira Belo.

2004

Dedico esta dissertação à Paula

Agradecimentos

Este mestrado foi parcialmente financiado pelo Departamento de Sistemas de Informação da Sonae Indústria Consultoria e Gestão. Agradeço a colaboração da empresa na realização deste objectivo pessoal e especialmente à Teresa Alves e ao Director do departamento Rufino Lopes. Agradeço também à empresa, e nomeadamente à Luz Ferreira, por ter autorizado a utilização do seu sítio *Web* corporativo como objecto de estudo.

Um agradecimento muito especial vai para o meu orientador, Professor Doutor Orlando Belo, pelo seu profissionalismo, pelos conselhos e pela dinâmica incutida na realização desta dissertação que só não avançou mais rápido devido a constrangimentos de ordem pessoal.

Agradeço a todos os meus familiares e amigos que, de forma directa ou indirecta, me deram o seu apoio na realização deste mestrado.

Agradeço à Anália Lourenço pela partilha de bibliografia e referências pois em muito facilitou a tarefa inicial de pesquisa.

Finalmente, agradeço à Paula por todo o seu apoio incondicional e paciência que sempre teve comigo.

Resumo

Sistemas de *Data Webhousing*: Análise, Desenho, Implementação e Exploração de Sistemas Reais

A *Web* tem-se tornado um dos espaços mais apelativos para as organizações como forma de divulgação das suas actividades, promoção dos seus produtos e serviços e desenvolvimento de actividades comerciais. Todavia, os visitantes de um sítio *Web* podem facilmente saltar para um sítio da concorrência caso não encontrem rapidamente aquilo que procuram, ou se tiverem qualquer outro motivo que não seja do seu agrado. Conhecer os visitantes e garantir que os produtos, serviços ou informação são aqueles que eles procuram é imperativo. É por isso que as organizações têm tentado analisar vários tipos de questões relacionadas, por exemplo, com a forma como os clientes procuram os produtos, onde abandonam o sítio e porquê, qual a frequência de visitas dos seus clientes, quais os produtos ou serviços que mais interesse despertaram nos visitantes, enfim tudo o que possa contribuir para a melhoria do sítio e para manter ou atrair novos clientes.

Todos os movimentos e selecções dos utilizadores de um sítio *Web* podem ser acompanhados através dos "cliques" que vão fazendo ao longo do seu processo de interacção com as diversas páginas *Web*. A esta sequência de "cliques" dá-se o nome de *clickstream*. Será a partir dos dados registados pelo servidor *Web* sobre as selecções do utilizador que se poderá iniciar o estudo das suas iterações e comportamento. Contudo, o registo mantido pelos servidores *Web* forma apenas

um esqueleto que terá de ser enriquecido com os registos dos vários componentes e sistemas que suportam o seu funcionamento. Este tipo de integração e conciliação de dados num único repositório é, tradicionalmente, feito no seio de um *Data Warehouse* que, pelo acréscimo dos dados de *clickstream*, se torna num *Data Webhouse*. Todo o processo de extracção, transformação e integração no *Data Webhouse* é, no entanto, dificultado pelo volume, incomplitude e heterogeneidade dos dados e pela própria tecnologia utilizada no ambiente *Web*.

Nesta dissertação, é apresentado e descrito um modelo dimensional para um *Data Webhouse* para análise de um sítio *Web* comercial. São estudadas e apresentadas algumas das suas fontes de dados bem como técnicas que podem ser utilizadas para eliminar ou reduzir os problemas existentes nos dados de *clickstream*. É descrito todo o desenvolvimento e implementação do processo de extracção, limpeza, transformação e integração de dados no *Data Webhouse* com especial relevo para as tarefas de *clickstream* - a identificação de utilizadores e agentes automáticos e a reconstrução de sessões. É apresentado o *Webuts – Web Usage Tracking Statistics*, um protótipo de um sistema de apoio à decisão para acompanhamento e análise estatística das actividades dos utilizadores de um sítio *Web* e onde se incorporam alguns dos elementos, técnicas, princípios e práticas descritas.

Palavras Chave: *Data Webhouse*, *Data Warehouse*, *Clickstream*, *Web*, logs de servidor Web, HTTP, identificação de utilizadores, identificação de sessões, heterogeneidade de dados, modelação dimensional

Abstract

Data Webhousing Systems: Analysis, Design, Implementation and Operation of Real Systems

The Web is becoming one of the most appealing environments for the many organisations as a means of promoting its businesses and activities as well as a commercialisation channel. However, a Web user can easily leave one organisation's Web site for its competitors if he doesn't find what he is looking for or if he finds something unpleasant on one organisation's site. To know the site's users and making sure that the products, services or information the site is providing is what the users want is nowadays a must. That is why many organisations have started to study how their web site users browse the site, where are they leaving the site and why, how frequently do their users return, what products and services are most appealing and, in general terms, everything that may be used to improve the Web site and attract new users.

Every user moves may be tracked by retaining the clicks selections they do on the different Web pages during their visit. This flow of clicks is now called clickstream. It is the data logged by the Web server on the user's selections that will enable the organisation to study their moves and behaviour. However, the Web server log only keeps the bare bones of the user's activity. This data will have to be enriched with data collected by other systems designed to provide the Web site with contents or additional functionalities. Traditionally, the gathering and integration of data from

heterogeneous data sources is done inside a Data Warehouse. By adding clickstream data to it we are creating a Data Webhouse. However, Web technology, the data volume, its heterogeneity and incompleteness will create difficulties in the process of extracting, transforming and loading data into the Data Webhouse.

In this document we present a dimensional model for a Data Webhouse whose purpose is to analyse a commercial Web site. Several data sources are presented and analised in detail. Some of the techniques used to eliminate or reduce clickstream data problems are also described. The Data Webhouse extraction, cleaning, transformation and loading process is described and special attention is paid to clickstream processing tasks such as user and robot identification and user session reconstruction. A new decision support system prototype, named Webuts - Web Usage Tracking Statistics, is presented. This system's purpose is to track and analyse a Web site users' moves and actitivities as well as generate some statistical data on the Web site operation. Its operation is based on a Data Webhouse and its development incorporated some of the elements, techniques and best practices studied and described.

Keywords: Data Webhouse, Data Warehouse, Clickstream, Web, Web server logs, HTTP, user identification, session identification, data heterogeneity, dimensional modelling

Índice

Índice de Figuras

Índice de Tabelas

Capítulo 1

Introdução

1.1 Contextualização

Desde a sua criação e até aos dias de hoje, o apelo a serviços e recursos da *Web* tem tido um aumento galopante. A generalidade das pessoas utiliza já a *Web* como meio para obtenção de ajuda para as suas actividades de lazer, diversão, estudo ou profissionais. O espaço *Web* é cruzado diariamente por inúmeros pedidos de informação provenientes dos mais variados sectores levados a cabo por uma população gigantesca de utilizadores. Não admira, pois, que um espaço como este seja tão apelativo para as organizações. Estas sentem cada vez mais a necessidade de se afirmarem neste meio ao qual recorrem cada vez mais pessoas, vistas como potenciais clientes, como primeiro ponto de pesquisa de informação. Eventualmente, se as empresas não estiverem presentes na *Web* poderão ser relegadas para segundo plano. Muitas pessoas já pensam que, se uma empresa não tem um sítio na *Web* é porque não tem significado comercial ou, simplesmente, não existe. É, pois, crucial ganhar um lugar de relevo no espaço da *Web*. No sentido de aproveitarem este mercado potencial, as empresas têm investido muito na sua presença nesse espaço, desenvolvendo sítios especializados para a divulgação das suas actividades, promoção dos seus produtos e serviços e desenvolvimento de actividades comerciais. Os seus sistemas de

informação têm vindo a ser gradualmente modificados de forma a poderem suportar essas actividades e disponibilizar todo um manancial de produtos e serviços que visem ir ao encontro dos interesses de quem os procura nesse espaço privilegiado. Assim, a primeira acção a fazer é atrair potenciais clientes. No mundo online, onde a oferta é imensa, há que tornar visível a presença da empresa. Uma das formas mais conhecidas para garantir essa visibilidade é fazer com que os motores de busca se interessem pelos sítios que desenvolveram, já que estes poderão ser, em muitos casos, a porta de entrada que maior número de visitantes atrai. Portanto, as empresas têm que preparar os seus sítios para que estes sejam facilmente indexáveis pelos motores de busca e desenvolver a própria organização no sentido de a preparar para servir esta franja tão particular de utilizadores.

Para as organizações que pretendam apenas deslocar para a *Web* uma base de clientes já captada por outros canais, seja por motivos de poupança de custos ou por qualquer outro, terão sempre de ter a noção de que os seus clientes não passarão a usar a *Web* somente porque vai facilitar a vida à organização. Se for mais fácil telefonar, mais seguro enviar um fax ou mais barato continuar com os processos actuais, então estes clientes não usarão o sítio *Web*. A empresa terá de simplificar todo o processo de iteração e de criar valor acrescentado para os seus clientes na *Web*. No entanto, a natureza predominantemente autónoma dos utilizadores de sítios e serviços de comércio electrónico tem colocado novos desafios às organizações. Os pedidos de páginas e serviços ocorrem a qualquer hora do dia e em qualquer altura do ano. Os utilizadores tanto podem inundar um sítio com um enorme número de pedidos como pura e simplesmente ignorá-lo. Na *Web* já serão raros os casos onde a vantagem de ser o primeiro ainda exista. Conhecer os visitantes e garantir que os produtos, serviços ou informação são aqueles que eles procuram é ainda mais imperativo que em qualquer outro canal. Enquanto que no mundo físico o factor conveniência é importante (ir à loja da esquina é conveniente porque fica perto), na *Web* este factor deixa de ser relevante. Os visitantes de um sítio de comércio electrónico podem facilmente saltar para um sítio da concorrência caso não encontrem rapidamente aquilo que procuram, ou se tiverem qualquer outro motivo que não seja do seu agrado. Qualquer organização que queira ter algum tipo de sucesso na *Web* terá que conhecer estes visitantes, já que serão eles os principais elementos que condicionarão a evolução dos seus sítios e as suas actividades de comércio electrónico, assim como a evolução da própria organização e dos seus modelos de negócio para a *Web*. Enquanto não conseguirem estabelecer os padrões de utilização dos seus sítios, é muito

provável que não consigam regular as suas actividades de gestão da forma mais conveniente e com a eficiência desejada. É por isso que as organizações têm tentado analisar vários tipos de questões relacionadas, por exemplo, com a forma como os clientes procuram os produtos, se abandonam o sítio e porquê, qual a frequência de visitas dos seus clientes, quais os produtos ou serviços que mais interesse despertaram nos visitantes, enfim tudo o que possa contribuir para a melhoria do sítio e para manter ou atrair novos clientes. Um cliente satisfeito poderá sempre voltar a comprar e será sempre um óptimo meio de divulgação do sítio entre os seus pares.

1.2 Motivação e Objectivos

O servidor *Web* assume um papel crucial pois passa a desempenhar um dos principais, senão o principal, pontos de contacto com os clientes. Será a partir dos dados por ele registados que se poderá iniciar o estudo das iterações com os visitantes. Todos os movimentos e selecções dos clientes de um sítio podem ser acompanhados através dos "cliques" - entenda-se selecção de apontadores *Web*, texto ou outros objectos gráficos localizados nas páginas - que vão fazendo ao longo do seu processo de interacção com as diversas páginas *Web* que o integram. Todos esses "cliques" ficam registados nos diversos componentes que suportam o processo de interacção com o utilizador. Por exemplo, de uma sequência de "cliques" poderá ficar registado algo como o seguinte:

Página A -> Página C -> Página B -> Página C -> Página F

De forma mais explícita, poderíamos traduzir, por exemplo, esta sequência pelo seguinte:

Início -> Catálogo de Produtos -> Registar -> Catálogo de Produtos -> Comprar

A esta sequência de "cliques" dá-se o nome de *clickstream*. Esta informação poderá, potencialmente, fornecer a mais detalhada informação que jamais uma organização conseguirá obter em qualquer contacto com os seus clientes.

Actualmente, já existem no mercado vários tipos de ferramentas, mais ou menos sofisticadas, mais ou menos dispendiosas, que dão uma ajuda na transformação e consequente exploração dos dados de *clickstream* registados pelos servidores *Web* das organizações. Todavia, estas ferramentas estão ainda limitadas no seu desempenho, facilidade de compreensão, profundidade de análise, validade e fiabilidade dos resultados produzidos. Por exemplo, não estão habilitadas, só por si, a efectuar correlações entre as vendas realizadas num sítio *Web* e as vendas efectuadas através de outros canais comerciais, ou simplesmente a medir o número de devoluções de encomendas feitas através do sítio *Web* por clientes do sexo feminino da classe média-alta. Existem situações em que há a necessidade de combinar a informação contida num *clickstream* com outra armazenada em outras fontes de informação. Assim, temos que, necessariamente, subir para um nível mais aplicacional e ir buscar a outros sistemas de informação os restantes dados necessários, sejam eles internos ou externos à organização [Lourenço et al. 03]. Este tipo de integração de dados, combinando informação proveniente de diversas fontes de informação autónomas e conciliando-a num único repositório de dados de sistema de suporte à decisão, é, tradicionalmente, feito num sistema de *Data Warehousing*. Ao se enriquecer este repositório com dados provenientes de *clickstreams* estamos a construir aquilo a que Kimball e Merz [KimballMerz00] chamaram um *Data Webhouse*. Num *Data Webhouse*, os dados de *clickstream* permitem realizar análises sobre o comportamento dos clientes durante as visitas que estes efectuam a um sítio *Web*. Os resultados desta análise dão, por sua vez, a possibilidade da organização corrigir e aperfeiçoar os seus sistemas por forma a retirar o máximo proveito de cada visita, tanto para o cliente como para si.

Temos então como objectivos desta dissertação de mestrado o estudo, análise, especificação, modelação e implementação de sistemas de *Data Webhousing* e mais concretamente em:

1. Elaborar sobre os requisitos e benefícios que podem levar uma organização a enveredar pela construção de um *Data Webhouse*.
2. Estudar a tecnologia e infra-estrutura utilizada em sítios *Web* e enumerar e descrever as diversas fontes de dados de um *Data Webhouse* com realce para as fontes de *clickstream*;
3. Especificar e apresentar um modelo dimensional passível de ser utilizado num *Data Webhouse* para um sítio *Web* transaccional.
4. Analisar os problemas resultantes da heterogeneidade dos dados de *clickstream* e apresentar medidas correctivas para os mesmos.

5. Estudar e detalhar o processo de Extracção, Transformação e Integração de dados no *Data Webhouse*, com especial destaque para as actividades de processamento impostas pelas características dos dados de *clickstream* e do ambiente *Web*.
6. Desenvolver um protótipo de um sistema de apoio à decisão, com base num *Data Webhouse*, para a análise estatística e acompanhamento das actividades dos visitantes de um sítio *Web* que demonstre a utilização de alguns dos elementos, técnicas, princípios e boas práticas estudadas e apresentadas.

Não é, contudo, objectivo desta dissertação de mestrado a apresentação de um documento que sirva de guia passo-a-passo para a gestão completa de um projecto de implementação de um *Data Webhouse.* Pretende-se sim que este documento possa estar lado a lado com esse guia.

1.3 Estrutura da Dissertação

Este documento encontra-se estruturado da seguinte forma :

- No capítulo dois são descritas algumas das razões e necessidades que levam ao surgimento de sistemas de *Data Webhousing*. É exposto o conceito de *Clickstream* e o valor potencial que este acarreta. São analisados os passos pelos quais passa um projecto de construção de um *Data Webhouse*. É explicado o princípio de funcionamento das comunicações do *Hyper Text Transport Protocol* (http) bem como introduzidos alguns elementos de rastreio de utilizadores que poderão auxiliar na análise do seu comportamento. É feita uma descição de uma possível arquitectura de suporte a um *Data Webhouse*.
- As fontes potenciais de dados de um *Webhouse* são descritas no capítulo três. É dado especial destaque à fonte principal dos dados de *clickstream*: os *logs* dos servidores *Web*.
- O capítulo quatro tem como objectivo a apresentação e descrição de um modelo dimensional para o *Data Webhouse*. Vários níveis de granulosidade serão descritos bem como algumas das possíveis variações aplicáveis ao modelo.
- No capítulo cinco é descrito todo o processo de extracção de dados. É efectuada uma análise aprofundada das fontes de dados mais significativas bem como o impacto que a sua heterogeneidade causa na construção de sistemas de *Data Webhouse*.

- No capítulo seis serão descritas as transformações pelas quais os dados destinados ao *Data Webhouse* terão de passar bem como o processo de integração nesta. São apresentados, também, alguns dos problemas e limitações resultantes da própria tecnologia usada em ambientes *Web*, que dificultam o processamento dos dados de *Clickstream* bem como algumas das técnicas passíveis de serem utilizadas para eliminar ou, pelo menos, minorar os seus efeitos.
- No capítulo sete são mencionadas algumas das técnicas a utilizar durante o processo de integração de dados no *Data Webhouse* bem como algumas das tarefas necessárias para a regular operação do mesmo. É ainda abordada a forma como Técnicas de Processamento Analítico podem ser usadas na exploração de *Data Webhouses* .
- O desenvolvimento do protótipo do *Webuts* - *Web User Tacking Statistics* é descrito no capítulo oito.
- Por fim, no capítulo nove, é efectuada uma súmula das principais vantagens e dificuldades no desenvolvimento e operação de sistemas de *Data Webhouse*, resultantes de todo o estudo e trabalhos efectuados durante este projecto de mestrado. São tiradas as conclusões desta dissertação bem como indicado alguns dos possíveis trabalhos a desenvolver futuramente.

Capítulo 2

Sistemas de *Data Webhouse*

2.1 Caracterização de um *Data Webhouse*

Existem na literatura científica vários exemplos da utilização de *Data Warehouses* para análise de dados de *clickstream*. Em [Bonchi et al. 01] o *Data Warehouse* é usado como suporte para o processo de mineração por forma a inferir algoritmos para caching inteligente de páginas *Web*. Em [Joshi et al. 99] e [Joshi et al. 03] é descrito o recurso a um *Data Warehouse* com armazenamento *Relational Online Analytical Processing* (ROLAP) para a descoberta, por processos de mineração, de padrões de acesso aos servidores *Web* tendo como única fonte de dados os *logs* de acessos. Também é descrita uma ferramenta de interrogação com um interface gráfico funcionando sobre a *Web*. Em [Zaiane et al. 98] é descrito o recurso ao uso de um *Data Warehouse* para integração de dados de acessos, fornecidos pelo servidor *Web*, com meta dados sobre a estrutura do sítio, e aplica técnicas de *Online Analytical Processing* (OLAP) e de mineração tendo em vista a descoberta de padrões de utilização do sítio *Web*. A aproximação dada em [Buchner et al. 99] é semelhante mas integra, também, dados de *marketing* e vendas como base para a personalização de serviços a prestar aos clientes. Em [KimballMerz00] é desenvolvida a mesma linha de pensamento e apresentada a definição de *Data Webhouse*. Este conceito é apresentado como podendo assumir

duas "personalidades". Por um lado, o *Data Webhouse* é a instanciação do clássico *Data Warehouse* enriquecido com dados de *clickstream* e, por outro, recorre às próprias tecnologias *Web* para a publicação da sua informação, ou seja, usar a *Web* para decidir sobre a *Web*. Em [Sweiger et al. 02] é dado seguimento à primeira "personalidade" do *Data Webhouse* sem, no entanto, usar o nome de *Data Webhouse*. Em [HuCercone04] é usada, seguindo a mesma nomenclatura de Kimball, um *Data Webhouse* como base para análises OLAP e processo de mineração. Sobre a segunda "personalidade" preconizada por Kimball, mover o *Data Warehouse* para a *Web*, é proposta em [Chen et al. 00] uma metodologia de implementação baseada em princípios que seguem a regra de 80/20, segundo a qual 80% das funcionalidades serão obtidas em apenas 20% do tempo total necessário para a implemetação completa do Data Warehouse.

Enquanto os sistemas operacionais, também conhecidos como *On-line Transaction Processing* (OLTP), dão o suporte ao modelo transaccional da organização, os sistemas de *Data Warehousing*, de uma forma geral, têm-se tornado os principais sistemas de suporte analítico das organizações modernas. Ambos são de vital importância para a organização mas, no que toca à sua construção e desenvolvimento, estes deverão seguir caminhos opostos. Nos sistemas operacionais é, tipicamente, utilizada uma modelação de dados baseada em entidades-relacionamento. Esta tem como objectivo directo o racionalizar do armazenamento de dados e agilizar o funcionamento transaccional da organização. Todavia, nos *Data Warehouses* o modelamento dimensional dos dados assume a predominância em alternativa à modelação por entidades-relacionamento (ER). Embora os dados constantes neste modelo tendam a ser os mesmos dos constantes no modelo ER dos sistemas operacionais, este é o melhor método de modelação para dados utilizados em sistemas de apoio à decisão [Kimball et al. 98]. A modelação dimensional estrutura os dados de uma forma mais compreensível e optimizada para interrogações. Recorre à utilização de tabelas de factos e dimensões por forma a fornecer, rápida e facilmente, as respostas para as perguntas mais prementes do negócio dando assim suporte ao processo de decisão da organização.

O projecto de implementação de um *Data Webhouse* poderá ser pensado coma a extensão de um projecto de implementação de um *Data Warehouse* clássico mas há diferenças que, sem dúvida, não podem ser esquecidas. No exemplo apresentado (Figura 2.1) podemos ver um esquema ilustrativo das diversas fases pelas quais passa um projecto de implementação de um *Data*

Webhouse [Sweiger et al. 02] que se traduz por uma especialização do proposto em [Kimball et al. 98] para projectos de *Data Warehouses* de cariz mais clássico.

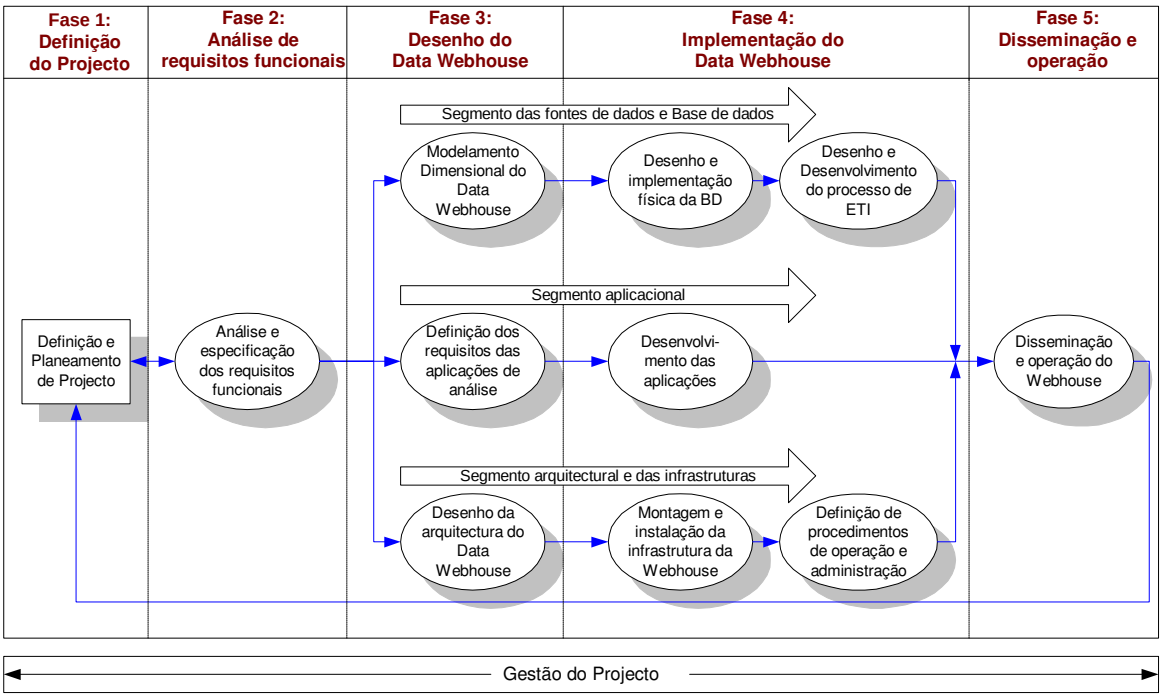

Figura 2.1 – Fases existentes num projecto de implementação de um *Data Webhouse*

Sem as pretender cobrir exaustivamente, podemos analisar quais os principais objectivos de cada uma destas fases.

Na **Definição do Projecto**, fase 1, deverá ser determinado exactamente qual, ou quais, são os objectivos da organização e definidos quais são os motivos para a realização do projecto. Estes podem ser diversos e serão claramente distintos em função do tipo de organização. Certamente uma Universidade que envereda por um projecto deste tipo não terá o objectivo primário de saber quais os produtos mais vendidos. De igual forma, uma instituição governamental que disponibilize ao contribuinte informação fiscal pela *Web* poderá ter objectivos distintos. O âmbito do projecto do *Data Webhouse* deverá ser nesta fase definido, nomeadamente, no que diz respeito às funcionalidades que serão incluídas e às que, explicitamente, serão excluídas. Aqui também se

deverá definir toda a equipa e processos necessários à implementação do projecto bem como a definição do seu plano de execução.

A fase 2, **Análise de Requisitos Funcionais**, tem como objectivo a análise e especificação dos requisitos funcionais do *Data Webhouse*. É uma fase, antes de mais, de conhecimento da organização e das necessidades de análise e de informação dos utilizadores finais. Será em função destes requisitos, e da análise mais profunda dos sistemas existentes, que deverão ser, então, descritas quais as mudanças, se necessárias, a ser efectuadas aos sítios *Web* bem como aos restantes sistemas fornecedores de dados. A prioridade dos resultados a serem produzidos pelo projecto deverá ser escalonada. Deverão ser feitas alterações ao âmbito, orçamento ou prazo em função deste escalonamento. No final, deverá ficar definido um plano de implementação detalhado que ditará o caminho a seguir nas fases seguintes.

A fase 3, **Desenho do *Data Webhouse***, decompõe-se em três segmentos, ou linhas de trabalho: o segmento das fontes de dados e Base de Dados, o segmento aplicacional e um terceiro segmento que trata da parte de arquitectura e infra-estrutura tecnológica. Estes mesmos segmentos serão continuados na fase seguinte. Nesta terceira fase pretende-se efectuar o modelamento dimensional dos *Data Marts* que constituirão o *Data Webhouse*. Esta modelação, conjuntamente com uma análise profunda das fontes de dados, tanto ao nível de sistemas que os originam como dos próprios dados em si, deverá ser feita por forma a poder elaborar uma estratégia para a selecção, extracção, transformação e carregamento dos dados. Deverão ser definidas e documentadas as funcionalidades pretendidas para os relatórios e aplicações de análise a serem utilizadas. Por fim, deverá ser identificado, avaliado e adquirido todo o software e hardware, ou serviços, necessários para a construção do *Data Webhouse*.

Quando se chega à fase de **Implementação do *Data Webhouse***, fase 4, toda a especificação já deve estar completa e ser conhecida por toda a equipa do projecto. Nesta fase, temos a montagem de toda a infra-estrutura do *Data Webhouse* (servidores, instalação de software de base, processos, etc.). É feito o desenvolvimento dos componentes de suporte à operação do *Data Webhouse* destacando-se, sem dúvida, os processos de extracção, transformação e integração de dados. São também desenvolvidas as aplicações que os utilizadores finais irão usar. Por fim, teremos a integração e teste de toda a infra-estrutura e fluxo dos dados no *Data Webhouse*.

A fase 5, **Disseminação e Operação**, é a fase onde o Webhouse é disponibilizado para operação. Convém que seja efectuado um planeamento de como deve ser feita esta "entrega": como e a quem dar formação, como e onde se irá instalar as aplicações de análise, se estas não forem feitas pela *Web*, e como será dado o suporte a estes utilizadores. A disseminação do *Data Webhouse* deverá ser faseada e começar por um pequeno grupo seleccionado de utilizadores. Deverá ser em função da reacção desses utilizadores que a disseminação aos restantes utilizadores será programada e, eventualmente, sujeitar a alterações os planos de formação e disseminação iniciais. Nesta fase, proceder-se-á à configuração final do ambiente produtivo e definir-se-á a forma de acomodar o crescimento do *Data Webhouse*. Bem como a criação dos processos e procedimentos do seu suporte e manutenção.

Num cenário de implementação de uma *Clickstream Data Mart*, ao contrário da implementação de outros *Data Marts*, onde são os administradores de bases de dados que controlam a generalidade das fontes de dados, a equipa de implementação necessita de trabalhar em parceria com administradores e programadores dos sítios *Web*. Este grupo de pessoas poderá ser o único que conseguirá fornecer a informação necessária para satisfazer os requisitos dos dados de *clickstream* do *Data Webhouse*. Será conveniente aos gestores do projecto de implementação familiarizarem-se com as tecnologias *Web*. De igual forma os programadores *Web* devem ficar ao corrente das necessidades específicas do *Data Webhouse* e de como as suas acções podem fazer evoluir, tanto pela positiva como pela negativa, toda a implementação.

Uma coisa que poderá surpreender a equipa de implementação do *Data Warehouse* é a quase constante mudança existente no ambiente da *Web*. Ao contrário dos sistemas operacionais que mudam a um ritmo mais estável e previsível, não é difícil encontrar situações onde se assiste a uma completa mudança tecnológica ou de estrutura e conteúdo de um sítio *Web*. O *Data Webhouse* tem de ser construído e preparado para lidar com esta mudança constante, caso contrário poderá estar condenado ao insucesso. Mais uma vez se salienta a importância da comunicação entre as equipas que implementam, ou mantêm, o *Data Webhouse* e os administradores e programadores do sítio *Web*.

Sobre os dados de *clickstream* enganem-se aqueles que julgam que estes são apenas mais uma fonte de dados. Estes dados estão espalhados por vários sistemas, com formatos heterogéneos e localizados em pontos potencialmente externos à organização. Nessa situação, o controlo da organização sobre estes dados é, no mínimo, limitado ou mesmo inexistente. Mais à frente voltaremos a este assunto.

Os dados de *clickstream* são, potencialmente, mais incompletos, mais expressivos e em maior número que em muitas outras fontes de dados. O processamento de largos volumes de dados de *clickstream* resultantes da interacção do visitante com os servidores *Web* trazem novos desafios [KrishnamurthyRexford98] nomeadamente a sobrecarga de processamento e problemas de integridade de dados. Basta lembrar os milhões de cliques que podem ser registados num dia, ou em algumas horas, num servidor *Web*. Pode bastar uma referência numa revista, num artigo de jornal ou numa notícia na televisão e mesmo o sítio *Web* com tráfego mais discreto poderá registar, subitamente, um nível de acessos completamente acima do esperado. O volume de dados a processar, nestas situações, dispara e todo o processo de extracção, transformação e integração tem de estar preparado para lidar com esse volume dentro da janela de oportunidade. Todo o processamento tem de estar optimizado ao máximo. Não é de todo recomendável ter de revisitar cada linha dos ficheiros com dados de *clickstream* durante o processo de ETI. Toda a informação deverá ser, idealmente, obtida e processada de uma só passagem. Não esquecer, porém, que os ficheiros de *logs* de acessos ao servidor *Web* incluem por vezes informação errada, ou inconsistente, que é preciso limpar ou omitir no processamento.

2.2 Rastreio de Utilizadores e Análise de Comportamento

2.2.1 Comunicação http

A comunicação entre o servidor *Web* e o navegador do visitante é efectuada com recurso ao protocolo HTTP [BernersLee et al. 96][Fielding et al. 99]. São várias as formas que podem fazer chegar um pedido HTTP a um sítio *Web*:

- Quando um utilizador escreve no seu navegador o URL do sítio *Web*.
- Quando um utilizador selecciona o URL da sua lista pessoal de apontadores.

- Quando um utilizador seleccionou um apontador existente numa página num outro sítio *Web*.
- Quando um utilizador seleccionou um apontador existente numa página dentro do sítio *Web*. Esse apontador pode ser de e para a mesma página.
- Quando um utilizador seleccionou num apontador incluído num documento de correio electrónico ou similar.
- Quando uma página, ou objecto, é incluída automaticamente por outra página.
- Quando agentes automáticos, tais como robots de indexação, efectuam pedidos de páginas *Web*.

O protocolo HTTP é essencialmente um protocolo de pedido e resposta. Quando o navegador do visitante faz um pedido por HTTP ao servidor *Web*, indicando qual o *Universal Resource Identificator* (URI) pretendido, o servidor responde a esse pedido e termina aí a relação com o navegador. As transacções HTTP não mantêm estado. O utilizador pode fechar o navegador, desligar-se e voltar a ligar-se à Internet e continuar a efectuar pedidos sobre as seguintes páginas no servidor. Necessita unicamente de indicar qual o URL da página pretendida e sem que o pedido dependa de qual, ou quais, as páginas visitadas anteriormente. A utilização de *cookies* foi originalmente proposta em [Netscape] e mais tarde redefinida em [KristolMontulli00] para solucionar o problema da falta de estado entre as interacções do visitante e o servidor *Web*. Essencialmente, uma *cookie* é uma variável cujo valor é uma sequência de caracteres. Obviamente que o valor e significado atribuído à *cookie*, ou *cookies*, pode depender totalmente da página *Web*, aplicação, ou servidor *Web* que a atribuiu. Uma *cookie* pode ser transiente, ou seja, mantida na memória do computador do visitante apenas enquanto o seu navegador estiver aberto, ou persistente, armazenada pelo navegador no disco do computador do visitante e disponível para futuras interacções com o sítio *Web*.

Na comunicação com um servidor *Web*, um utilizador faz, normalmente, um pedido HTTP de cada vez. Todavia, este pedido pode dar origem a múltiplas iterações entre o navegador e o servidor. Uma página *Web* é, na maior parte das vezes, composta por múltiplos objectos – imagens, ficheiros de som ou vídeo, rotinas em Javascript – individualmente endereçáveis por URLs distintos. O navegador ao compor a página *Web* no computador do utilizador irá automaticamente

fazer pedidos HTTP individuais para cada um dos componentes que integra a página. Seguindo uma comunicação entre o navegador de um visitante e o servidor *Web* (Figura 2.2) podemos ver os cabeçalhos HTTP transmitidos. O significado das variáveis transmitidas nos cabeçalhos está descrita em [Fielding et al. 99]. Pela análise destes cabeçalhos, constatamos que o cliente iniciou a visita com um pedido da página "inicio.asp". Na resposta, o servidor enviou a página pedida e simultaneamente atribuiu uma *cookie* de sessão ao visitante. De seguida, o visitante seleccionou um apontador existente na página "inicio.asp" e o navegador efectuou o pedido da página "/liverpool/liverpool.asp". A *cookie* anteriormente atribuída pelo servidor passou também a ser transmitida nos pedidos do visitante embebida no cabeçalho do protocolo HTTP. O servidor respondeu enviando a página pedida. Na página "liverpool.asp" estava uma referência a uma imagem. O navegador automaticamente faz um novo pedido ao servidor para obter essa imagem. O servidor responde com o envio da imagem e fica assim completa a página "liverpool.asp".

2.2.2 Elementos de Rastreio

De acordo com [W3C99] uma sessão, ou visita, é composta pelo conjunto de actividades efectuadas por um utilizador desde o momento em que este entra num sítio *Web* até ao momento que ele o deixa. [Joshi et al. 00] definem uma sessão de um utilizador como o conjunto de acesso originários do mesmo endereço do tipo *Internet Protocol* (IP) tais que o intervalo de tempo entre estes acessos não ultrapasse um valor pré-determinado. Esta segunda definição não é válida para utilizadores que, por exemplo, estão por detrás de *firewalls* ou *proxies* que substituem o endereço IP original do utilizador pelo seu próprio.

Relembra-se que o pedido de uma página *Web* origina vários pedidos aos objectos que a compõem. Se considerarmos que várias páginas poderão ser visitadas, então o conjunto de pedidos HTTP ainda aumenta mais. Se considerarmos agora os múltiplos visitantes que um sítio pode ter simultaneamente então, potencialmente, o número de pedidos HTTP é imenso.

Figura 2.2 - Comunicação HTTP entre cliente e servidor *Web*

Há que encontrar então um elemento que permita identificar todos os pedidos HTTP que fazem parte da mesma sessão. Embora esta temática seja desenvolvida mais à frente podemos adiantar alguns candidatos que podem contribuir para esta função:

- Endereço IP da origem do pedido mais identificação do cliente HTTP.
- *Cookies* de sessão transientes atribuídas automaticamente pelo servidor *Web* ou programaticamente pelas aplicações *Web*.
- *Cookies* persistentes atribuídas na maior parte dos casos programaticamente, embora também o possam ser automaticamente pelo servidor *Web*.
- Variável com identificador de sessão passada como parâmetro no URI do pedido HTTP.
- Uso de autenticação do utilizador pelo servidor *Web* para o acesso a páginas seguras.

Ao identificarmos a sessão podemos começar a estudar o comportamento dos visitantes. Todavia, não sabemos quem são esses visitantes. Este tipo de informação pode não ser precisa para sítios onde não seja necessário distinguir individualmente cada visitante mas unicamente analisar as visitas como um todo. Se o sítio for do tipo transaccional, então será, certamente, necessário identificar cada visitante. Esta é, sem dúvida, uma das tarefas mais difíceis de conseguir. O conhecimento que se pode obter sobre cada visitante é variável sendo na maioria das situações impossível saber quem foi a pessoa que efectuou uma determinada visita. Há, contudo, algumas informações que se podem obter sobre o visitante mesmo não conhecendo a sua identidade, nomeadamente:

- País de origem do pedido HTTP.
- Fornecedor de acesso à Internet do visitante ou empresa detentora do endereço IP usado.
- Navegador e configurações do computador tais como resolução gráfica, hora local.

Em sítios transaccionais o registo dos utilizadores é, sem dúvida, um dos requisitos já que é necessária informação que permita, pelo menos, a realização da transacção comercial e identifique qual o local para a entrega dos produtos ou serviços transaccionados. Neste caso, o utilizador efectua a sua autenticação no sítio *Web* em cada visita, sendo facilitado todo o processo de identificação de sessão e do utilizador.

2.2.3 Análise Comportamental

Uma das forças impulsionadoras ao desenvolvimento e implementação do *Data Webhouse*, é sem dúvida o *Customer Relationship Management* (CRM) ou, posto numa perspectiva da *Web*, segundo [Sweiger et al. 02], *Electronic Relationship Management* (ERM): a tentativa de obtenção do maior grau de satisfação e lealdade por parte dos clientes conseguindo simultaneamente um aumento dos benefícios para a organização, sejam eles financeiros ou outros.

Num *Data Webhouse*, os dados provenientes de *clickstream*s permitem realizar análises comportamentais dos clientes nas visitas que estes efectuam a um sítio *Web*. Esta análise poderá ajudar a compreender melhor os comportamentos do cliente perante as diversas páginas *Web* que se lhe apresentam e, porventura, inferir comportamentos de mais alto nível resultantes de cada visita:

- Se uma compra teve sucesso ou não.
- Se a informação procurada foi encontrada.
- Se o visitante saiu do sítio *Web* e ou cancelou o pedido a determinada página.
- Se visitante entrou num sítio *Web* por engano.

Foi constatado em [MoeFader01] que clientes com uma crescente frequência de visitas a um sítio *Web* tendem a ser clientes mais valiosos, tanto em valores absolutos como percentuais, já que têm uma maior probabilidade de efectuar uma compra. Clientes cuja frequência de visitas esteja a diminuir terão, pelo contrário, uma menor probabilidade de comprar. Será, pois, vantajoso para a organização tornar o sítio o mais apelativo possível por forma a fazer com que os visitantes retornem. Isto pode ser alcançado através do recurso a técnicas de configuração e personalização.

Por **configuração** entenda-se alteração efectuadas pelo próprio visitante, em opções disponibilizadas pelo sítio *Web*, de acordo com as suas preferências. Esta configuração pode no entanto ser semi-automática baseada no perfil de um dado utilizador. A **personalização** é automática e recorre à análise do comportamento do cliente, ao perfil do cliente ou eventualmente aos perfis de outros grupos de clientes semelhantes. Estudando os dados de visitas passadas, deverá ser possível induzir quais os hábitos e preferências dos clientes. Estes mesmos hábitos e preferências poderão então ser utilizados como base para suporte de futuras visitas

nomeadamente através sistemas de recomendação [Schafer et al. 01], reajustamentos no design, sugestões de navegação ou selecção de conteúdos. A personalização e configuração permitirão proporcionar aos clientes uma experiência mais agradável, prestando sítios com conteúdos que lhes sejam mais relevantes e permitindo-lhes encontrar mais rapidamente aquilo que procuram - isto traduz-se na prática em menos "cliques" e menos tempo despendido. Estes factores contribuirão positivamente para induzir o tão desejado retorno destes clientes.

2.2.4 Arquitectura de um *Data Webhouse*

Podemos dizer que um *Data Webhouse* veio aumentar as já existentes exigências colocadas sobre o tradicional *Data Warehouse*.

- **Actualidade e validade da informação** - O ritmo das tomadas de decisões empresariais tem aumentado significativamente mas no ambiente *Web* estas poderão ter de ser quase imediatas. Exige-se que a informação sobre a situação da organização seja disponibilizada em tempo real, ou quase real, para dar suporte a esta tomada de decisões. A Webhouse tem de dar resposta a este requisito já que o recurso ao relatório do dia anterior deixou de ser suficiente em muitas organizações.
- **Capacidade de processamento** - Com a criação de um canal transaccional adicional pela *Web*, ou mesmo como um canal exclusivo, começou-se a registar e analisar todos os cliques dos visitantes. O volume de dados explodiu, com servidores *Web* de diversas organizações a registarem milhões de acessos nos seus ficheiros de *log*. O Webhouse tem de ser capaz de processar estes dados.
- **Rapidez de resposta** – Se as aplicações *Web* necessitarem de informações processadas pelo Webhouse e estas não estiverem disponíveis numa questão de segundos, isso pode significar o princípio do fim do sítio *Web*. O utilizador pode simplesmente desistir de esperar e saltar facilmente para outros sítios.

No exemplo apresesentado (Figura 2.3) podemos ver um exemplo de uma possível arquitectura para um *Data Webhouse* num ambiente transaccional via *Web* derivada do modelo proposto em [KimballMerz00]. São muitas as variações possíveis desta arquitectura e esta deverá ser planeada em função da organização e objectivos do *Data Webhouse*. A arquitectura descrita é direccionada

para organizações com um Webhouse para suporte ao canal de vendas via *Web*. A informação é disponibilizada por um servidor *Web* Aplicacional, tanto para utilizadores internos como remotos. A autenticação e autorização de acessos ao *Data Webhouse* são geridas por um servidor *Ligth-Weight Distributed Access Protocol* (LDAP). Neste servidor a cada utilizador é associado um, ou mais, perfis. Pressupondo um acesso aos objectos no Webhouse controlado por perfis simplifica-se, assim, toda a gestão da segurança do *Data Webhouse*. Os acessos dos utilizadores remotos são feitos pela Internet com recurso a uma rede privada virtual onde toda a informação segue encriptada.

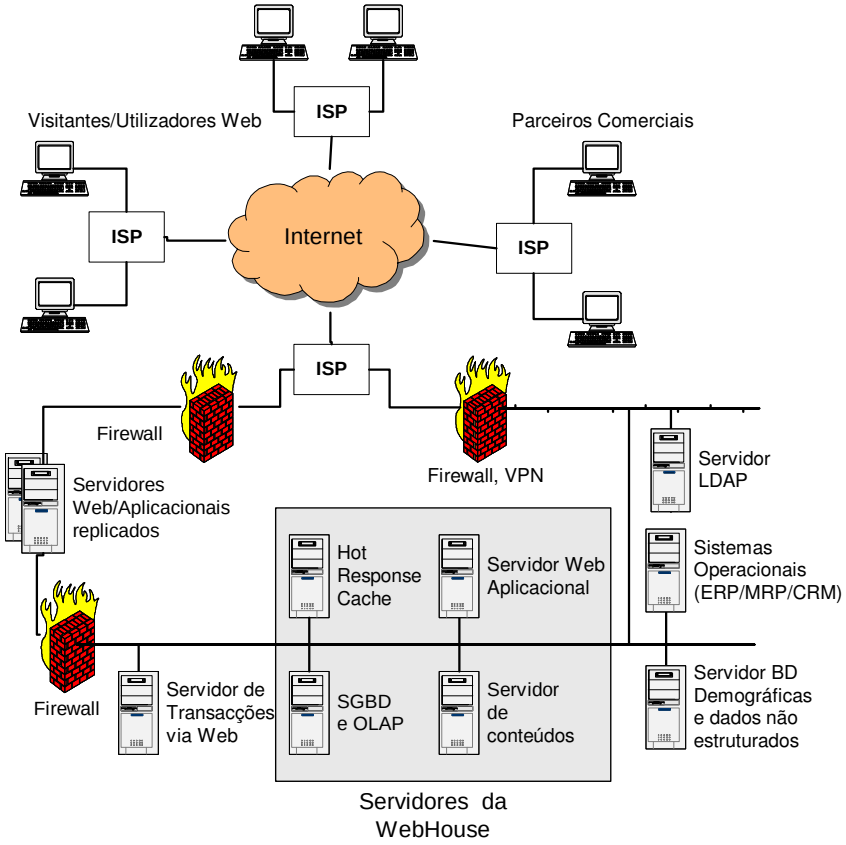

Figura 2.3 – Arquitectura de um *Data Webhouse*

Os servidores *Web* públicos, funcionando com um mecanismo de balanceamento de cargas, não guardam estado e gerem apenas a camada de apresentação para as aplicações transaccionais. O estado, ou dados específicos destas transacções, é guardado num servidor de transacções

dedicado. Poder-se-á optar por ter estas mesmas transacções nos sistemas operacionais da organização. No entanto, esta situação poderá não ser possível em casos onde seja necessário o uso de sistemas específicos para, por exemplo, a recepção e verificação de pagamentos com cartão de crédito ou o sistema transaccional exigir optimizações específicas para os pedidos via *Web*.

A aposta para os servidores do *Data Webhouse* é feita no paralelismo com divisão de tarefas. Se um só servidor for utilizado, este apenas pode ser expandido até um certo limite. Podemos, no entanto, desde o início, apostar e preparar todos os processos para serem distribuídos e funcionarem em paralelo permitindo assim, e em caso de necessidade, a expansão da capacidade do sistema pelo adicionar de mais máquinas.

O *Hot response cache* é o servidor responsável pelo armazenamento de respostas pré-processadas. A informação aqui armazenada tanto pode ser preparada para o servidor *Web* público como para o servidor *Web* Aplicacional do *Data Webhouse*, como por exemplo:

- Mensagens de boas vindas aos clientes via *Web*.
- Promoções preparadas em função do perfil do cliente.
- Informações frequentemente pedidas por clientes ou parceiros comerciais (últimos dez pagamentos, últimas dez encomendas, últimas dez entregas, etc.).
- Análises específicas com estudo de evolução ao longo de períodos de tempo.
- Análises sumárias resultantes de agregações comuns (em função do produto, do cliente, do período).

Estas informações seriam carregadas neste servidor através de processamento em bloco. Desta forma, ajudam a aumentar a rapidez das respostas. As aplicações devem ser preparadas desde o início para tirar partido deste servidor e, caso este não possa fornecer as respostas, então devem reverter para um modo de operação por omissão e, por exemplo, apresentar uma saudação genérica ou então avisar o utilizador de que o seu pedido tardará algum tempo a ser servido. Este servidor poderá reduzir bastante a carga posta no *Sistema de Gestão da Base de Dados* (SGBD) do *Data Webhouse*. Há que garantir, contudo, que ele próprio não é um factor de atraso nas respostas em situações de alto débito de informação. Grande cuidado deve ser posto na configuração dos sistemas de leitura/escrita em disco. O mesmo é válido para os restantes

servidores. Estes deverão recorrer a sistemas que permitam leituras em paralelo da informação armazenada nos discos, seja pela utilização de sistemas de RAID, seja pela distribuição de cargas por diversos canais e placas controladoras de discos. No caso específico do servidor, ou servidores, onde corre o SGBD a capacidade e velocidade de escrita deverão ser dimensionadas por forma a conseguir que toda a informação que necessita de ser carregada durante o processo de ETI o seja dentro da janela de oportunidade.

Capítulo 3

Fontes de Dados do *Data Webhouse*

Embora as fontes de dados para o *Data Webhouse* possam ser as mesmas usadas em *Data Warehouses* mais clássicas, tais como informação sobre clientes, informações sobre produtos ou serviços, informação sobre a data e hora. A grande diferença está na nova fonte de dados surgida com o ambiente *Web*: dados de *clickstream*.

A escolha das fontes dependerá essencialmente do objecto primordial de análise que é atribuído ao *Data Webhouse*. Em organizações que efectuem transacções de cariz comercial pelo sítio *Web* será necessário recolher dados dos sistemas operacionais que lhes dão suporte. No entanto, este tipo de informação já não é necessário se estivermos a falar de um *Data Webhouse* para uma instituição que apenas sustém um sítio *Web* como meio de divulgação de informação.

No exemplo ilustrado (Figura 3.1) podemos ver algumas das possíveis fontes de dados. Uma das características, não muito vista em projectos de *Data Warehousing*, é o recurso a fontes de dados externas à organização. O controlo sobre essas fontes é, no mínimo, limitado, ou mesmo inexistente, já que estes ficam espalhados pelos diversos operadores de redes e serviços que permeiam e possibilitam a comunicação do navegador do visitante com o sítio *Web*. No capítulo 5 serão analisados com detalhe alguns dos problemas que estas fontes de dados externas poderão criar.

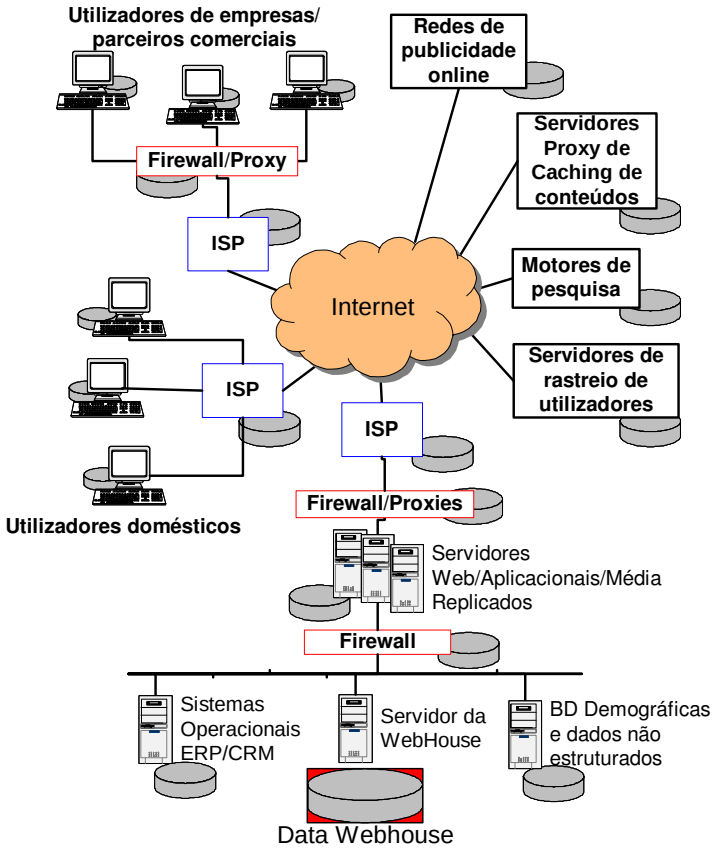

Figura 3.1 – Fontes de dados de um *Data Webhouse*

3.1 Formatos *standard* de Logs

Os *logs* de servidores *Web* são a principal fonte dos dados de *clickstream* para um *Data Webhouse*. É nestes ficheiros que ficam registados os pedidos HTTP efectuados pelos navegadores dos visitantes aos servidores *Web*. A informação a disponibilizar na *Webhouse* dependerá em grande parte do detalhe, qualidade e quantidade de dados registado pelos servidores *Web* nestes ficheiros, principalmente se estes forem a única fonte de dados.

Existem muitos formatos de *log* com um nível de detalhe variável. Quais os formatos disponibilizados e que nível de detalhe se poderá obter está dependente do servidor *Web*. Existem, no entanto, alguns formatos *standard* criados pela *National Center for Supercomputing Applications* (NCSA) e pela World Wid Web Consortium (W3C). Actualmente os mais significativos são:

- *NCSA Common Log Format (CLF)*.
- *NCSA Extended Common Log Format (ECLF)*.
- *W3C Extended Log Format*.

3.1.1 NCSA Common Log Format

O *Common Log Format* definido em [Luotonen95] e [NCSAHTTPd], surgiu pela primeira vez com o servidor HTTPD desenvolvido pela NCSA. Este formato de *log* tentou pôr fim aos formatos proprietários utilizados pelos vários servidores *Web* e tornou-se no primeiro formato *standard* de facto (Tabela 3.1). Com o surgimento deste *standard* ficou simplificada a tarefa de criação de programas de análises estatísticas para os vários servidores *Web* pois estes passaram a ter de trabalhar apenas com um único formato.

Dos vários formatos *standard* existentes, este é o que menor detalhe regista, mas quase todos os servidores *Web* o suportam. Vejamos então um exemplo no *Common Log Format*:

```
10.32.100.1 - pgonzalez [10/Oct/2003:13:55:36 +0100] "GET
/catalogo.html HTTP/1.1" 200 2326
```

Passando a explicar o seu significado temos então:

```
10.32.100.1
```

Este é o endereço IP do cliente (*host* remoto) que efectuou o pedido ao servidor. É possível efectuar uma configuração onde o IP seja traduzido para o nome. Esse processo de resolução de endereços é, no entanto, pouco recomendável, já que pode afectar significativamente o desempenho do servidor. Há que ter em atenção que este IP poderá não ser da máquina de onde

o utilizador se está a conectar mas sim de um proxy ou de uma *Firewall* que poderá estar entre o cliente e o servidor.

Campo	***CLF***	***ECLF***	***Descrição***
remotehost	√	√	Este é o endereço IP do cliente, caso a resolução de nomes dados pelos *Domain Name Services* (DNS) não esteja activa, que efectuou o pedido ao servidor.
Ident	√	√	Identidade remota fornecida pelo cliente do pedido HTTP e atribuída pelo processo identd no lado do cliente, segundo processo de autenticação definido no RFC931 [StJohns85] entretanto substituído pelo RCF1431 [StJohns93].
authuser	√	√	O nome ou código com que o utilizador se autenticou no servidor *Web*.
Date	√	√	Data e hora em que o servidor terminou de servir o pedido .
request	√	√	Este campo indica o pedido feito ao servidor. Este campo é delimitado por aspas.
status	√	√	O código de estado HTTP retornado para o cliente resultante da acção efectuada. A lista completa dos códigos de estado HTTP 1.1 está especificada na secção 10 do RFC2616 [Fielding et al. 99].
bytes	√	√	Número de *bytes* enviados pelo servidor ao cliente HTTP. Não inclui o tamanho dos cabeçalhos HTTP.
referer		√	Este é o valor transmitido na variável Referer do cabeçalho HTTP (embora se escreva Referrer a especificação técnica do HTTP menciona a variável como sendo Referer). O seu valor indica por quem o pedido do cliente diz ter sido referenciado.
user-agent		√	O cliente HTTP utilizado no pedido. Normalmente identifica o navegador *Web*.

Tabela 3.1 - Campos do *Common Log Format* e *Extended Common Log Format*

Olhando para o segundo campo apresentado notamos a existência do seguinte caracter:

```
-
```

O hífen presente logo ao seguir ao endereço IP indica a ausência da informação. Caso estivesse presente, este campo seria a identificação do utilizador com autenticação efectuada pelo processo *identd* a correr do lado do cliente. É, no entanto, uma informação pouco fiável e não deverá ser usada excepto em redes internas com níveis de segurança bem definidos e controlados.

```
pgonzalez
```

Este é o identificador do utilizador autenticado através do processo de HTTP. Se o estado HTTP do pedido for o 401 então este identificador não deverá ser usado pois o utilizador não está ainda autenticado. Se não for necessária uma palavra chave para aceder ao documento pedido, então este campo, tal como o anterior, tomará o valor de "-".

```
[10/Oct/2001:13:55:36 +0100]
```

A data e a hora em que o servidor *Web* terminou de servir o pedido. O formato seguido é o seguinte:

```
[dia/mês/ano:hora:minuto:segundo zona]
      dia = 2*dígitos
      mês = 3*letras
      ano = 4*dígitos
      hora = 2*dígitos
      minuto = 2*dígitos
      segundo = 2*dígitos
      zona = (`+' | `-') 4*dígitos
```

```
"GET /catalogo.html HTTP/1.1"
```

Este campo indica o pedido feito ao servidor. Neste caso foi usado o método GET para aceder á página `/catalogo.html`. O protocolo usado foi o `HTTP/1.0`.

```
200
```

Este é o estado HTTP, resultante da execução do pedido, retornado ao cliente. Códigos começados por 2 indicam uma resposta com sucesso, começados com 3 indicam um redireccionamento, começados com 4 indicam um erro causado pelo cliente e começados com 5 indicam um erro do servidor.

```
2326
```

Este último é o tamanho do objecto enviado na resposta ao cliente HTTP. Neste valor não está considerado o tamanho dos cabeçalhos HTTP. Se nada foi retornado para o cliente então teremos também o "-".

3.1.2 NCSA Extended Common Log Format

O *Extended Common Log Format*, também disponibilizado no servidor *Web* da NCSA [NCSAHTTPd], resultou de uma extensão do *Common Log Format* pelo acréscimo dos campos: referenciador e agente HTTP (Tabela 3.1). Este formato é também conhecido por *Combined Log Format* porque passou a incluir num único ficheiro a informação que estava espalhada por três ficheiros distintos: o *log* de acessos, o *log* dos referenciadores e o *log* dos agentes HTTP. Vejamos o seguinte exemplo:

```
10.32.100.1 - pgonzalez [10/Oct/2003:13:58:36 +0100] "GET
/catalogo.html HTTP/1.1" 200 2326
"http://www.someorg.biz/inicio.html" "Mozilla/4.08 [en] (Win98; I
;Nav)"
```

Dissecando o seu conteúdo temos então :

```
"http://www.someorg.biz/inicio.html"
```

Este campo contém o valor da variável *Referer* incluída no cabeçalho do pedido HTTP dirigido ao servidor *Web*. Indica qual a página que por quem o pedido HTTP diz ter sido referenciado. Ou seja, a página que continha um apontador, ou uma inclusão, para o objecto pedido. O

referenciador tanto pode ser uma página de um sítio externo como de uma página existente no próprio sítio *Web*. Caso o pedido não tenha sido referenciado, ou seja, o URI do objecto tenha sido escrito directamente na barra de endereços ou escolhido a partir da lista de apontadores do navegador, então seria o símbolo do hífen que apareceria entre aspas ("-").

```
"Mozilla/4.08 [en] (Win98; I ;Nav)"
```

Este campo indica o valor da variável *User-Agent* do pedido HTTP dirigido ao servidor *Web*. Contém, normalmente, a versão do navegador que fez o pedido. Este valor pode, no entanto, ser manipulado [Opera] dissimulando assim qual o cliente HTTP que realmente efectuou o pedido.

3.1.3 W3C Extended Log Format

Este formato foi desenvolvido pelo *World Wide Web Consortium* (W3C) tendo vista a criação de um *standard* que pudesse satisfazer as necessidades de clientes, servidores e *proxies* [HallamBehlendorf96a] [HallamBehlendorf96b]. A especificação deste formato ainda está em fase de rascunho não sendo ainda considerada como uma recomendação da W3C.

Directiva	***Significado***
Version	A versão do formato utilizado.
Fields	Indica quais os campos registados no ficheiro.
Software	Identifica o software que gerou o ficheiro de *log*.
Start-Date	A data e hora a que o registo no *log* começou.
End-Date	A data e hora em que terminou o registo do *log*.
Date	Data e hora de registo da directiva.
GMT-Offset	Desvio de tempo em relação à hora *Greenwich Mean Time* (GMT). Esta directiva é obrigatória caso os tempos usados sejam registados na hora local e afecta apenas o campo da hora.
Remark	Comentários em texto livre.

Tabela 3.2 - Directivas do *W3C Extended Log Format*

Todos os ficheiros de *log* neste formato são auto-identificadores através do recurso a um cabeçalho que precede todas as outras entradas. Este cabeçalho contém directivas (Tabela 3.2), identificadas pelo caracter cardinal (#) no início de cada linha, que indicam, entre outros, quais os tipos de dados registados bem como a versão do formato usado.

Neste formato, os campos são separados por espaços, embora também seja possível a utilização de caracteres de tabulação. Os espaços contidos no URI são codificados por forma a garantir que não são interpretados como separadores de campos. Para campos seleccionados mas para os quais não existe informação o caracter hífen (-) é colocado na respectiva posição do campo.

Prefixos	***Significado***
c	Cliente.
s	Servidor.
r	Remoto.
cs	Cliente para servidor.
sc	Servidor para cliente.
sr	Servidor local para servidor remoto, prefixo usado por proxies.
rs	Servidor remoto para servidor local, prefixo usado por proxies.
x	Prefixo específico de aplicações.

Tabela 3.3 - Prefixos de campos do *W3C Extended Log Format*

Os campos registados neste formato podem ser prefixados (Tabela 3.3) por forma a indicar com quem a informação está relacionada ou qual o sentido da informação. O formato W3C é um formato flexível permitindo variar o nível de detalhe que se quer registar. São vários os campos possíveis de serem utilizados (Tabela 3.4) bem como os prefixos aplicáveis [HallamBehlendorf96a] [HallamBehlendorf96b]. Sempre que existe uma alteração do número de campos, é registado um novo cabeçalho com as respectivas directivas indicando quais os campos que passam a ser registados. A especificação da W3C também refere a possibilidade de utilizar campos que sumariam a utilização por parte de ferramentas de análise. Este tipo de campos não é de interesse para o *Data Webhouse* já que escondem o detalhe que afinal se procura e, como tal, não serão aqui descritos.

Campo	***Prefixos aplicáveis***	***Descrição***
date	Nenhum	Data GMT em que o pedido HTTP terminou de ser servido.
time	Nenhum	Hora em que o pedido HTTP terminou de ser servido A hora é GMT a não ser que a directiva GMT-Offset seja especificada.
time-taken	nenhum	Tempo, em segundos, que demorou a servir o pedido.
bytes	nenhum	Número de *bytes* transferidos.
cached	nenhum	Indica se objecto estava em *cache* ou não. O valor zero indica que não.
ip	c; s; r	Endereço e porta IP. No caso do servidor a porta será omissa caso se trate de uma porta *standard* , exemplo: porta 80 para HTTP, 443 para *Secure Hyper Text Transport Protocol* (HTTPS).
dns	c; s; r	Nome do computador no formato retornado pelos servidores de DNS.
status	sc; rs	O código de estado HTTP resultante da acção, descrito na secção 10 do RFC2616 [Fielding et al. 99].
comment	sc; rs	Comentário complementar descritivo do estado HTTP retornado.
method	cs; sr	Acção executada, por exemplo um GET, um HEAD ou um POST. A lista completa dos métodos para o HTTP 1.0 pode ser consultada no RFC1945 [BernersLee et al. 96] e para o HTTP 1.1 no RFC2616 [Fielding et al. 99].
uri	cs; sr	*Uniform Resource Identifier* (URI) do objecto manipulado segundo a sintaxe especificada nos RFCs 1630 [BernersLee94], 1738 [BernersLee et al. 94], 1808 [FieldingIrvine95] e 2396 [BernersLee et al. 98].
uri-stem	cs; sr	Identificação do nome e caminho dentro do servidor *Web* para o recurso pedido, por exemplo uma página HTML, ou uma imagem. Exclui o nome do servidor e a os parâmetros do URI.
uri-query	cs; sr	A componente de pares variável e valor incluída no URI após o caracter "?".
auth-id	c; s; r	Contém o identificador, normalmente o código de utilizador, usado para efeitos de autorização.
(nome)	variável	O conteúdo da variável identificada por nome transmitida no cabeçalho do pedido e resposta HTTP. As variáveis *standard* estão definidas no

		RFC2616 [Fielding et al. 99] e RCF2965 [KristolMontulli00] . Podemos ter, entre outros, os seguintes campos : ***cs(From)*** – Email do visitante, pouco usado. ***cs(Referer)*** - indica por quem o pedido do cliente diz ter sido referenciado. ***cs(User-agent)*** - contém o cliente HTTP utilizado no pedido. Normalmente identifica o navegador *Web*. ***cs(Cookie)*** - identificadores utilizados para a troca de informação de estado. ***sc(Server)*** – contém informação sobre o software usado para servir o pedido.

Tabela 3.4 – Descrição dos campos do *W3C Extended Log Format*

Podemos observar o exemplo de uma entrada no formato *W3C Extended Log File Format* gerada, neste caso, por um servidor *proxy* de *cache- Oracle 9i Application Server Web Cache* :

```
#Version: 1.0
#GMT-Offset: -0500
#Software: Oracle9iAS Web Cache/2.0.0.2.0
#Start-Date: 2002-10-24 00:00:15
#Fields: c-ip c-dns c-auth-id date time cs-method cs-uri sc-status
bytes cs(Cookie) cs(Referrer) time-taken cs(User-Agent)
#Date: 2002-10-24 00:00:15
10.32.37.2 pcmfr03.someorg.org jsmith 2002-10-24 00:00:18 GET
/admin/images/office.jpg 200 350
"Server_Webcache_pool=1443321748;ORA_UOM_AGID=%2fMP%2f8M7%3f%3fDh3V
H"
"http://www.someorg.org/nl/about.html" 6 "Mozilla/4.5 [en] (WinNT;
I)"
```

Neste exemplo, podemos observar as várias directivas presentes e de notar especialmente a directiva *Fields* que nos indica quais os campos, e respectiva sequência, que vão aparecer no *log*.

```
10.32.37.2
```

Este é o endereço IP do cliente (*host* remoto) que efectuou o pedido ao servidor. Há que ter em atenção que este IP poderá não ser da máquina de onde o utilizador se está a conectar mas sim de um proxy ou de uma *Firewall* que poderá estar entre o cliente e o servidor.

```
pcmfr03.someorg.org
```

Nome do cliente (*host* remoto) que efectuou o pedido ao servidor. Nem sempre a tradução de endereços IP em nomes está activa pois esse processo de resolução de endereços pode afectar significativamente o desempenho do servidor. Se esse processo de resolução de endereços IP em nomes não estivesse activo então teríamos o símbolo do hífen (-) em vez do nome. Há que ter em atenção que este nome, tal como o endereço IP, poderá não ser da máquina de onde o utilizador se está a conectar mas sim de um proxy ou de uma *Firewall* que poderá estar entre o cliente e o servidor.

```
jsmith
```

Este é o identificador do utilizador autenticado através do processo de HTTP. Se o estado HTTP do pedido for o 401 então este identificador não deverá ser usado pois o utilizador não está ainda autenticado. Se não for necessária, ou não estiver presente, uma palavra chave para aceder ao documento pedido então este campo tomará o valor de hífen (-).

```
2001-10-24
```

A data em que o servidor terminou de servir o pedido. Esta é data GMT.

```
00:00:18
```

A hora em que o servidor terminou de servir o pedido. Ter em atenção que a directiva GMT-Offset toma o valor de -0500. Isto quer dizer que a hora especificada no *log* corresponde à hora local. A hora GMT seria então 05:00:18.

```
GET
```

Este campo indica o método utilizado no pedido feito ao servidor. Neste caso foi usado o método GET. Outros métodos comuns incluem o HEAD e o POST.

```
/admin/images/office.jpg
```

Parte do URI que identifica dentro do sítio *Web* qual o objecto pedido.

```
200
```

Este é o estado HTTP, resultante da execução do pedido, retornado pelo servidor ao cliente. Códigos começados por 2 indicam uma resposta com sucesso, começados com 3 indicam um redireccionamento, começados com 4 indicam um erro causado pelo cliente e começados com 5 indicam um erro do servidor.

```
350
```

Este é o tamanho, em *bytes*, retornado pelo servidor ao cliente do pedido. Neste valor não está considerado o tamanho dos cabeçalhos HTTP. Se nada foi retornado para o cliente então teremos também o hífen (-).

```
"Server_Webcache_pool=1443321748; ORA_UOM_AGID
=%2fMP%2f8M7%3f%3fDh3VO"
```

Cookies transmitidas pelo cliente ao servidor. São pares de variáveis e valores. A separação dos diversos pares é feita pelo caracter ponto-e-vírgula (;).

```
"http://www.someorg.org/nl/about.html"
```

Este campo contém o valor da variável *Referer* incluída no cabeçalho do pedido HTTP dirigido ao servidor *Web*. Indica qual a página por quem o pedido HTTP diz ter sido referenciado. Ou seja, a página que continha um apontador, ou uma inclusão, para o objecto pedido. O referenciador tanto pode ser uma página de um sítio externo como de uma página existente no próprio sítio *Web*. Caso o pedido não tenha sido referenciado, ou seja, o URI do objecto tenha sido escrito directamente na barra de endereços ou escolhido a partir da lista de apontadores do navegador, então seria o símbolo do hífen que apareceria entre aspas ("-").

```
6
```

Tempo, em segundos, que o servidor demorou a servir o pedido.

```
"Mozilla/4.5 [en] (WinNT; I)"
```

Este campo indica o valor da variável *User-Agent* do pedido HTTP transmitido pelo cliente ao servidor *Web*. Contém, normalmente, a versão do navegador que fez o pedido. Este valor pode, no entanto, ser manipulado [Opera] dissimulando assim qual o cliente HTTP que realmente efectuou o pedido.

3.2 Logs do Servidor *Web Apache*

O *Apache* é um servidor *Web* de código aberto cujo desenvolvimento é suportado pela *Apache Software Foundation*. Em Setembro de 2003, segundo [Netcraft03], o *Apache* era o servidor *Web* mais popular a nível mundial com cerca de 28 milhões de servidores, ou seja, 65% do número total de servidores activos.

O servidor HTTP da *Apache* disponibiliza um mecanismo de *logging* bastante completo e flexível [ApacheHTTP]. São vários os *logs* existentes entre os quais destaca-se o de erros e o de acessos. No primeiro o servidor regista os erros ocorridos durante o processamento de pedidos HTTP bem

como informação complementar de diagnóstico. É, no entanto, no segundo tipo de *log*, o *log* de acessos, que a nossa atenção se centra. O detalhe pretendido no ficheiro é especificado através de directivas (Tabela 3.5) dadas no ficheiro de configuração do servidor.

Directiva	***Descrição***
%%	O símbolo de percentagem (*Apache* 2.0.44 e versões superiores).
%a	Endereço IP remoto.
%A	Endereço IP local.
%b	Número de *bytes* enviados, excluindo o tamanho de cabeçalhos HTTP. Se o *Common Log Format* for utilizado então neste caso o hífen (-) será usado quando o número de *bytes* enviados for zero.
%B	Número de *bytes* enviados, excluindo o tamanho de cabeçalhos HTTP.
%{nome}C	Conteúdo da *cookie* identificada por nome.
%D	Tempo, em micro-segundos, que demorou a servir o pedido HTTP.
%{NOME}e	Conteúdo da variável de ambiente identificada por NOME.
%f	Nome do ficheiro .
%h	Endereço, ou nome, do computador remoto.
%H	O protocolo utilizado no pedido.
%{nome}i	O conteúdo da variável identificada por nome constante no cabeçalho HTTP do pedido enviado ao servidor. As variáveis *standard* estão definidas no RFC2616 [Fielding et al. 99] e RCF2965 [KristolMontulli00]. Podemos ter, entre outros, os seguintes: ***%(Referer)i*** - Indica por quem o pedido do cliente diz ter sido referenciado. ***%(User-agent)i*** - Contém o cliente HTTP utilizado no pedido. Normalmente identifica o navegador *Web*.
%I	Número de *bytes*, incluindo o tamanho dos cabeçalhos HTTP, recebidos pelo servidor. Este valor não pode ser zero.
%l	Identidade remota fornecida pelo cliente do pedido HTTP e atribuída pelo processo identd no lado do cliente, segundo processo de autenticação definido no RFC931[StJohns85] entretanto substituído pelo RCF1431 [StJohns93].
%m	Acção executada, por exemplo um GET, um HEAD ou um POST. A lista completa dos métodos para o HTTP 1.0 pode ser consultada no RFC1945 [BernersLee et al. 96] e

	para o HTTP 1.1 no RFC2616 [Fielding et al. 99].
%{nome}n	Conteúdo da nota identificada por nome transmitida por outro módulo.
%{nome}o	O conteúdo da variável identificada por nome constante no cabeçalho HTTP da resposta enviada pelo servidor ao cliente. As variáveis *standard* estão definidas no RFC2616 [Fielding et al. 99] e RCF2965 [KristolMontulli00]. Podemos ter, entre outros, os seguintes: ***%{Server}o*** – contém informação sobre o software usado para servir o pedido. ***%{Location}o*** – contém informação sobre o novo URI para onde o pedido original do cliente é redireccionado.
%O	Número de *bytes*, incluindo o tamanho dos cabeçalhos HTTP, enviados pelo servidor *Web*. Este valor não pode ser zero.
%p	O número da porta IP onde o servidor está a servir o pedido HTTP.
%P	O número do processo filho que serviu o pedido HTTP.
%{formato}P	O número do processo filho, ou *thread*, que serviu o pedido HTTP. Os valores que `formato` pode tomar são do `process id` (`pid`) ou `thread id` (tid) (*Apache* 2.0.46 e versões superiores).
%q	A componente de pares de variáveis e valores incluída no URI após o caracter "?" ou um texto vazio, caso este não exista.
%r	Identificação, composta por caminho e nome, do recurso pedido, por exemplo, uma página HTML, ou uma imagem.
%s	O código de estado HTTP resultante da acção, descrito na secção 10 do RFC2616 [Fielding et al. 99]. Para pedidos que foram internamente redireccionados este é o estado do pedido original. Caso se pretenda o estado resultante após o redireccionamento então devemos utilizar `%>s`
%t	Data e hora, segundo o formato Inglês, ou seja, aquele usado pelo *Common Log Format* em que o servidor terminou de servir o pedido.
%{formato}t	Data e hora, seguindo a especificação dada por formato. Esta especificação de formato deverá ser segundo a descrita nas páginas do manual do `strftime(3)`.
%T	Tempo, em segundos, que demorou a servir o pedido
%u	O nome ou código com que o utilizador se autenticou no servidor *Web*. Este não é válido se o estado HTTP for igual a 401.
%v	Nome do servidor *Web*, e porta IP se diferente de porta 80, especificado pela directiva

	ServerName no ficheiro de configuração do *Apache*. Este nome poderá ser diferente daquele que se obteria se se fizesse a resolução do endereço IP do servidor *Web* no nome retornado por um servidor de DNS.
%V	Nome do servidor *Web*, e porta IP se diferente de porta 80. Este nome pode ser obtido de três formas diferentes. A forma utilizada é ditada pelo valor da directiva de configuração `UseCanonicalName`.
%X	Estado da conexão quando o pedido fica completo: X = conexão abortada antes de completar o pedido. + = conexão pode ser mantida após a resposta ter sido enviada. - = conexão é fechada após a resposta ter sido enviada.

Tabela 3.5 - Directivas de configuração do mecanismo de *log* do *Apache* v2

Convém também dizer que o registo dos valores identificados por estas directivas pode ser condicionado pelo código de estado HTTP resultante. Podíamos, por exemplo, registar o nome do utilizador dado por `%u` se o código HTTP não fosse igual a 401. Para esse caso teríamos então de usar `%!401u`.

Com este conjunto de directivas podem ser facilmente configurados *logs* de acesso de múltiplos formatos. Por exemplo, a configuração que deveria ser feita para obtermos o NCSA *Common Log Format (CLF)* seria a seguinte:

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog logs/access.log common
```

Para obtermos o NCSA *Extended Common Log Format* (também conhecido por *Combined Log Format*) utilizaríamos a seguinte configuração:

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-
agent}i\"" combined
CustomLog log/acces.log combined
```

É possível também separar o registo de acessos por múltiplos *logs*: Referenciadores num ficheiro, pedidos noutro, etc. No entanto, esta separação não tráz vantagens para a construção do *Data Webhouse*. Bem pelo contrário, apenas iria trazer uma maior complexidade ao processo de extracção de dados já que haveria a necessidade de juntar todos os registos referentes ao mesmo pedido HTTP. O *Apache* também pode criar *logs* condicionais, ou seja, excluir o registo de certos pedidos em função dos valores de variáveis de ambiente. Mais uma vez, este tipo de *logs* não é do interesse do *Data Webhouse* já que se pretende ter o máximo detalhe registado.

Embora bastante completo, o servidor *Web Apache* não tem, de raiz, a capacidade de criar *logs* de acessos no formato *W3C Extended Log Format*. Não é que o mesmo tipo de dados não consiga ser registado mas, as directivas específicas do formato da W3C, obrigatórias no início do ficheiro que identificam o conteúdo do mesmo, não conseguem ser aí colocadas pelo *Apache*.

3.3 Logs do Servidor *Web* Microsoft IIS

Segundo a mesma sondagem citada para o servidor *Apache* [Netcraft03], o *Microsoft Internet Information Services* (IIS) ocupava a segunda posição de servidor *Web* mais popular. Com um número a rondar os 10 milhões, detinha uma quota de cerca de 24% do número total de servidores *Web*.

O servidor *Web* IIS é distribuído conjuntamente com o sistema operativo da Microsoft. São vários os formatos de *log* que este servidor *Web* pode utilizar[MicrosoftIIS][MicrosoftIIS6RK03]:

- *Microsoft W3C Extended Log Format*.
- *IIS Log File Format*.
- *NCSA Common Log Format*.
- *Log Open Database Connectivity* (ODBC).
- *Log* Binário Centralizado.

No entanto, nem todos os formatos de *log* estão disponíveis e variam em função da versão tanto do sistema operativo como do próprio servidor *Web*. Por exemplo, a capacidade de registo de *log* via ODBC directamente para uma base de dados não está disponível com o servidor IIS disponível

no *Microsoft Windows 2000 Professional*. O servidor *Web* IIS, v5.x e v6, também permitem a criação de *logs* personalizados através do recurso a objectos COM que disponibilizam um conjunto de métodos ao servidor. A forma de implementar este tipo de *logs* não será objecto de análise neste documento e pode ser consultada em [MicrosoftIIS6RK03].

3.3.1 Microsoft W3C Extended Log Format

Este formato (Tabela 3.6) é baseado no formato proposto pela W3C tendo, no entanto, algumas diferenças. É adicionado, por exemplo, um campo de estado, `sc-win32-status`, específico dos sistemas operativos Microsoft Windows. De notar também que o campo `time-taken` é expresso em milisegundos e não em segundos como a W3C define. O IIS também pode colocar no *log* informação que a Microsoft designa por *Process Accounting*. Esta informação extra permite-nos obter informação muito específica sobre o sistema operativo e processos da máquina onde o servidor *Web* corre. Este tipo de informação não será, no entanto, descrita neste documento mas aconselha-se uma leitura mais atenta da documentação do IIS se o detalhe pretendido no *Data Webhouse* for de cariz bastante técnico. Para os campos seleccionados mas para os quais não haja informação será usado o símbolo do hífen (-). Se o valor de um campo contiver um caracter que não seja imprimível então este será substituído pelo caracter mais (+). Podemos ver um pequeno exemplo do *log* gerado neste formato:

```
#Software: Microsoft Internet Information Services 5.0
#Version: 1.0
#Date: 2003-03-06 00:31:35
#Fields: date time c-ip cs-username s-ip s-port cs-method cs-uri-stem cs-
uri-query sc-status sc-win32-status sc-bytes cs-bytes time-taken cs-host
cs(User-Agent) cs(Cookie) cs(Referer)
2003-03-06 06:47:35 213.205.83.49 - 213.205.83.50 80 GET
/hab/images/bt_ir_ES.gif - 304 0 164 822 15 biz.someorg.org
Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT+5.0)
MSCSProf=95385A1F52DEA1A22;+CampaignHist=21;+ASPSESSIONIDCQBRTQCD=EGGDJBLB
OGIJHGBBLBLINAAD http://biz.someorg.org/hab/index.asp?market=es
```

Campo	***Descrição***
date	Data em que o pedido terminou de ser servido (GMT).
time	Hora em que o pedido terminou de ser servido (GMT).
c-ip	Endereço IP do cliente.
cs-username	Nome do utilizador.
s-sitename	Nome do site/serviço no servidor.
s-computername	Nome do servidor onde o ficheiro de *log* foi gerado.
s-ip	Endereço IP do servidor onde o ficheiro de *log* foi gerado.
s-port	Porta IP onde o navegador do cliente ligou (normalmente 80 para HTTP ou 443 para https).
cs-method	Acção executada, por exemplo um GET, um HEAD ou um POST. A lista completa dos métodos para o HTTP 1.0 pode ser consultada no RFC1945 [BernersLee et al. 96] e para o HTTP 1.1 no RFC2616 [Fielding et al. 99].
cs-uri-stem	O caminho dentro do servidor *Web* para o recurso acedido, por exemplo uma página HTML, ou uma imagem.
cs-uri-query	A componente de parâmetros incluída no URI após o caracter "?".
sc-status	O código de estado HTTP resultante da acção, descrito na secção 10 do RFC2616 [Fielding et al. 99].
sc-win32-status	O código de estado, utilizado pelo Microsoft Windows, resultante da acção. O significado deste código pode ser visto através da utilização do comando "`net helpmsg n`", onde n é o valor registado no *log*, na linha de comando do sistema operativo.
sc-bytes	Número de *bytes* enviados pelo servidor.
cs-bytes	Número de *bytes* recebidos pelo servidor.
time-taken	O tempo que a acção demorou a ser executada, em milisegundos.
cs-version	A versão do protocolo usada pelo navegador do cliente (por exemplo http1.0 ou HTTP 1.1).
cs-host	O nome, indicado pelo cliente HTTP, do sírtio *Web* que deverá servir o pedido.
cs(user-agent)	O cliente HTTP utilizado no pedido. Normalmente identifica o navegador *Web.*
cs(cookie)	O conteúdo das *cookies* enviadas pelo cliente HTTP, se estas existirem.
cs(referrer)	Este valor indica por quem o pedido do cliente diz ter sido referenciado.
sc-substatus	Sub-estado HTTP definido pelo IIS. (IIS versão 6.0 apenas) [MicrosoftKB318380].

Tabela 3.6 – Descrição dos campos do *log* W3C gerado pelo IIS

3.3.2 IIS Log File Format

Este formato específico do IIS é um formato com um número de campos fixo (Tabela 3.7). Regista, no entanto, mais alguma informação que aquela disponibilizada pelo *NCSA Common Log Format*. Neste formato, os campos são separados no ficheiro de *log* por vírgulas e a hora registada é sempre a hora local.

Campo	***Descrição***
Endereço IP	Endereço IP do cliente.
Nome do utilizador	Nome do utilizador.
Data	Data em que o pedido terminou de ser servido.
Hora	Hora em que o pedido terminou de ser servido.
Serviço e Instancia	Nome do site/serviço no servidor.
Nome do computador	Nome do servidor onde o ficheiro de *log* foi gerado.
Endereço IP do servidor	Endereço IP do servidor *Web* ao qual o ficheiro de *log* diz respeito.
Tempo demorado	O tempo que a acção demorou a ser executada, em milisegundos.
Bytes enviados	Número de *bytes* enviados pelo servidor.
Bytes recebidos	Número de *bytes* recebidos pelo servidor.
Estado http	O código de estado HTTP resultante da acção, descrito na secção 10 do RFC2616 [Fielding et al. 99].
Estado do Windows	O código de estado, utilizado pelo Microsoft Windows, resultante da acção. O significado deste código pode ser visto através da utilização do comando "`net helpmsg n`", onde n é o valor registado no *log*, na linha de comando do sistema operativo.
Método	Acção executada, por exemplo um GET, um HEAD ou um POST. A lista completa dos métodos para o HTTP 1.0 pode ser consultada no RFC1945 [BernersLee et al. 96] e para o HTTP 1.1 no RFC2616 [Fielding et al. 99].
Uri acedido	O caminho no servidor para o recurso acedido, por exemplo, uma página HTML, ou uma imagem.
Parâmetros	Parâmetros que são passados para uma página ou programa .

Tabela 3.7 – Informação constante no *IIS Log File Format*

Podemos ver de seguida um exemplo de entrada registada com este formato de *log*:

```
10.32.100.1,   -,   10/26/2003,   21:33:49,   W3SVC1,   EBPORTATIL,
10.32.100.3,   110,   498,   533,   200,   0,   GET,   /Default.htm,
sec=20.33%20;view=20,
```

Convêm referir que a data segue o formato americano: `Mês/Dia/Ano` (ex.:`10/26/2003`). Este formato usa dois dígitos para representar o ano para datas menores ou iguais que 1999 e quatro dígitos para datas posteriores. Há que ter em atenção que se o URI contiver vírgulas, estas poderão ser erradamente interpretadas como separadores de campos por parte dos programas analisadores de *logs*.

3.3.3 NCSA Common Log Format

Este formato é essencialmente o formato *standard* já descrito anteriormente. O nome remoto do utilizador neste caso é sempre indicado com o caracter hífen (-). Se um campo contiver um caracter não imprimível então este será substituído pelo caracter mais (+). A hora especificada é sempre a hora local conjuntamente com a especificação do desvio relativamente à hora GTM.

3.3.4 Log ODBC

Este é mais um formato com um número de campos fixo (Tabela 3.8). Os dados são registados directamente numa base de dados acessível via ODBC.

De notar que, face ao limite em número de caracteres, os campos `Target` e `Parameters` poderão não ser capazes de armazenar toda a informação proveniente de URIs, ou parâmetros, bastante longos.

Campo	*Tipo de dados*	*Descrição*
ClientHost	varchar(255)	Endereço IP do cliente.
UserName	varchar(255)	Nome do utilizador.
LogTime	datetime	Data e hora em que o pedido terminou de ser servido.
Service	varchar(255)	Nome do site/serviço no servidor.
Machine	varchar(255)	Nome do computador onde o ficheiro de *log* foi gerado.
ServerIP	varchar(50)	Endereço IP do servidor *Web* ao qual o ficheiro de *log* diz respeito.
ProcessingTime	integer	O tempo que a acção demorou a ser executada, em segundos.
BytesRecvd	integer	Número de *bytes* recebidos pelo servidor.
BytesSent	integer	Número de *bytes* enviados pelo servidor.
ServiceStatus	integer	O código de estado HTTP resultante da acção , descrito na secção 10 do RFC2616 [Fielding et al. 99].
Win32Status	integer	O código de estado, utilizado pelo Windows, resultante da acção. O significado deste código pode ser visto através da utilização do comando "*net helpmsg n*", onde n é o valor registado no *log*, na linha de comando do sistema operativo.
Operation	varchar(255)	Acção executada, por exemplo um GET, um HEAD ou um POST. A lista completa dos métodos para o HTTP 1.0 pode ser consultada no RFC1945 [BernersLee et al. 96] e para o HTTP 1.1 no RFC2616 [Fielding et al. 99].
Target	varchar(255)	O caminho no servidor para o recurso acedido, por exemplo uma página HTML, ou uma imagem.
Parameters	varchar(255)	Parâmetros que são passados para uma página ou programa.

Tabela 3.8 - Campos para o efectuar o *log* via ODBC

3.3.5 Log Binário Centralizado

Este é o formato de *log* mais recente da Microsoft e está disponível apenas na versão 6 do IIS. Ao contrário de recorrer a um ficheiro de texto, como na maioria dos formatos anteriormente

descritos, recorre a um ficheiro num formato binário para registo dos pedidos HTTP efectuados ao servidor *Web*. Este ficheiro único pode ser usado para centralizar o registo de todos os pedidos HTTP efectuados aos múltiplos sítios *Web* alojados num único servidor *Web*.

Campo	***Descrição***
Data	Data em que o pedido terminou de ser servido.
Hora	Hora em que o pedido terminou de ser servido.
Endereço IP	Endereço IP do cliente.
Nome do utilizador	Nome do utilizador.
Serviço e Instancia	Nome do site/serviço no servidor.
Nome do servidor	Nome do servidor *Web* onde o ficheiro de *log* foi gerado.
Endereço IP do servidor	Endereço IP do servidor *Web* ao qual o ficheiro de *log* diz respeito.
Porta	Porta IP onde o navegador do cliente ligou (normalmente 80 para HTTP ou 443 para https).
Método	Acção executada, por exemplo um GET, um HEAD ou um POST. A lista completa dos métodos para o HTTP 1.0 pode ser consultada no RFC1945 [BernersLee et al. 96] e para o HTTP 1.1 no RFC2616 [Fielding et al. 99].
URI Stem	O caminho dentro do servidor *Web* para o recurso acedido, por exemplo uma página HTML, ou uma imagem.
URI Query	A componente de parâmetros incluída no URI após o caracter "?".
Estado http	O código de estado HTTP resultante da acção, descrito na secção 10 do RFC2616 [Fielding et al. 99].
Estado do Windows	O código de estado, utilizado pelo Windows, resultante da acção. O significado deste código pode ser visto através da utilização do comando "`net helpmsg n`", onde n é o valor registado no *log*, na linha de comando do sistema operativo.
Bytes enviados	Número de *bytes* enviados pelo servidor.
Bytes recebidos	Número de *bytes* recebidos pelo servidor.
Tempo demorado	O tempo que a acção demorou a ser executada, em milisegundos.
Versão do protocolo	A versão do protocolo usada pelo navegador cliente (por exemplo http1.0 ou HTTP 1.1).
Sub-estado	Sub-estado HTTP definido pelo IIS. (IIS versão 6.0 apenas).

Tabela 3.9 - Campo usados no formato binário do Microsoft IIS

Em termos de dados registados (Tabela 3.9) estes são semelhantes ao formato *Microsoft W3C Extended Log Format.* A estrutura interna deste ficheiro binário está descrita em [MicrosoftIIS6RK03].

De notar que, ao contrário do formato *Microsoft W3C Extended Log Format,* não é incluída qualquer informação sobre *cookies*, referenciadores, agente do utilizador e nome do computador cliente. Sem a informação de *cookies* poderá não ser possível reconstruir a sessão do visitante. Sem a informação dos referenciadores também não será possível saber se um visitante chegou ao sítio *Web* por indicação de um motor de pesquisa ou um parceiro comercial.

3.4 Logs de Servidores *Proxy* de *Cache*

Os *proxies* podem ser usados em vários locais (Figura 3.2) e desempenhar várias funções:

- **Autenticação de acessos**: podendo, ou não, funcionar conjuntamente com *Firewalls* na identificação de quem acede à Internet a partir de uma rede privada.
- **Autorização de acessos**: pela filtragem sobre os *sítios* ou páginas a que os utilizadores podem aceder.
- **Ocultação de identidade**: utilizado quando não se pretende que os servidores *Web* conheçam, em termos de identificadores possíveis na rede, a verdadeira identidade do visitante. A utilizações deste tipo de *proxys* torna anónimas todas as iterações com o servidor *Web*, sendo praticamente impossível efectuar o rastreio de quem os usa.
- ***Cache* de conteúdos**: permitindo servir mais rapidamente os utilizadores e ao mesmo tempo permitir uma redução do tráfego na rede.

É precisamente esta última função que poderá ter interesse para o *Data Webhouse*, não numa perspectiva dos acessos efectuados por utilizadores internos para o exterior mas precisamente no sentido inverso.

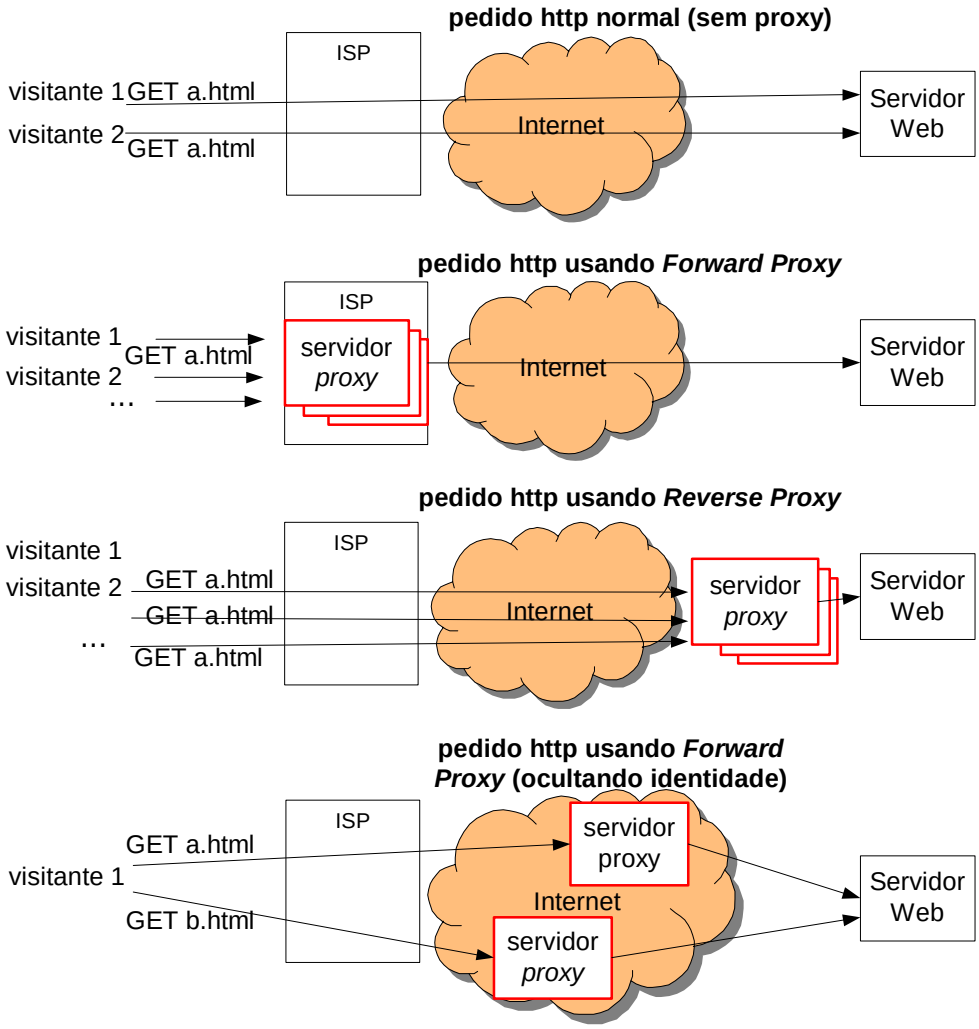

Figura 3.2 – Utilizações de Proxies

Um *Internet Service Provider* (ISP) pode usar *proxies* para efectuar *cache* das páginas e conteúdos mais acedidos pelos seus clientes e desta forma poupar em largura de banda (*forward proxy*). De igual forma uma empresa também poderá usar *proxies* à frente dos servidores *Web* (*reverse proxy*) para efectuar *cache* das páginas e conteúdos mais acedidos e aliviar a carga dos servidores *Web* principais.

Campo	***Descrição***
Tempo	Identificador temporal no formato Unix (número de segundos passados desde 1/Janeiro/1970) e com precisão de milisegundos.
Duração	O tempo que o pedido tardou a ser servida em milisegundos.
Endereço do Cliente	Endereço IP, ou nome caso a resolução de nomes esteja activa no servidor, do cliente que efectuou o pedido.
Código de estado	Código de estado resultante da execução do pedido. Composto por códigos internos do Squid (TCP_HIT, TCP_MISS, etc.) conjuntamente com um subconjunto dos códigos de estado HTTP definidos no RFC2616 [Fielding et al. 99] mais alguns definidos para funcionarem com *Web*DAV.
Bytes	Tamanho em *bytes* da resposta enviada ao cliente do pedido, inclui tamanho dos cabeçalhos, e seja ela de sucesso ou de erro.
método	Acção executada, por exemplo um GET, um HEAD ou um POST. A lista completa dos métodos para o HTTP 1.0 pode ser consultada no RFC1945 [BernersLee et al. 96] e para o HTTP 1.1 no RFC2616 [Fielding et al. 99]. Alguns mais são também reconhecidos para o *WebDAV*.
URL	URL do objecto. Parâmetros eventualmente passados não são registados.
Identidade remota	Identidade remota fornecida pelo cliente do pedido HTTP e atribuída pelo processo *identd* no lado do cliente, segundo processo de autenticação definido no RFC931 [StJohns85] entretanto substituído pelo RCF1431 [StJohns93]. Se não atribuída ou não registada então no *log* será colocado um hífen (-) no seu lugar.
Código hierárquico	Uma descrição de como e quando o objecto a ser servido foi obtido bem como qual o servidor que o serviu.
Tipo	Valor da variável `Content-type`**,** quando presente**,** do cabeçalho HTTP da resposta.
Cabeçalhos do pedido http	Em modo de debug apenas. Todas as variáveis do cabeçalho do pedido HTTP podem ser registadas no *log*.
***Cabeçalhos da resposta* HTTP**	Em modo de debug apenas. Todas as variáveis do cabeçalho da resposta HTTP podem ser registadas no *log*.

Tabela 3.10 – Campos do formato de *log* nativo do *proxy* Squid v2

O servidor *proxy* de *cache* pode explicitamente servir documentos pré-computados que demorariam demasiado tempo a serem servidos pelo servidor *Web* em tempo real. Pode também ser contratado externamente o serviço de *cache* a fornecedores com ligações de alta velocidade

aos segmentos principais da Internet. Esta é uma forma de colocar o conteúdo mais perto dos utilizadores e diminuir a latência nas comunicações permitindo que estes sejam servidos mais rapidamente. Este tipo de servidores poderá gerar *logs* complementares aos do servidor *Web* e poderá, por exemplo, dar informações sobre pedidos HTTP que foram servidos pela *cache* dos *proxy* e que nunca chegaram ao servidor *Web*.

Na descrição do formato de *log W3C Extended Log Format* podemos ver um exemplo de um *log* de um servidor de *Reverse Proxy*: Oracle 9i Application Server *Web Cache*. Vejamos um outro exemplo de um popular servidor de *proxy* de *cache* de código livre – Squid. São vários os tipo de *logs* que o Squid pode gerar [SquidLog]. É, no entanto, o *log* de acessos que mais nos interessa.

O Squid pode gerar este *log* tanto no *NCSA Common Log Format*, já descrito, como no seu formato próprio. Embora o *NCSA Common Log Format* seja reconhecido por um maior número de ferramentas e porventura de mais fácil integração no *Data Webhouse*, o formato de *log* nativo do Squid é mais rico em detalhes. O formato nativo do Squid (Tabela 3.10) consiste num ficheiro de texto com, pelo menos, 10 campos separados por um ou mais espaços.

Eis um pequeno exemplo de algumas entradas no formato nativo do Squid v2.

```
1020816267.836  193  61.87.2.67  TCP_MISS/302  644  GET  http://home-
l3.tiscali.nl/%7Eti017329/honeyz01.jpg    -    DIRECT/195.241.76.80
text/html
1020816304.598   55   226.90.141.125   TCP_REFRESH_HIT/304   203   GET
http://ar.atwola.com/content/B0/0/H7pTL2Luf0_kw3xmlj8W1sns8a9RRNke8
_SAqLzKBa609jmULHVa8jgFKtiL69KXCWvLTQ4eKHG6BVFfpwz9J2_nwVlARAAN-
pkCJqF1Tww$/aol - DIRECT/152.163.226.185 -
1020816320.249  130  134.202.51.180  TCP_REFRESH_MISS/200  2226  GET
http://disney.go.com/globalmedia/pardonourdust/background.gif     -
DIRECT/63.70.47.83 image/gif
```

3.5 Firewalls

As *Firewalls* são normalmente instaladas em pontos onde é visível todo o tráfego que sai e entra na rede. Podem, ou não, funcionar integradas com servidores *proxy*. Os seus *logs* podem ser usados na *Webhouse* para complementar a informação proveniente dos *logs* dos servidores *Web* com informação sobre emails enviados, acessos via ftp ou outros protocolos cuja análise seja relevante. Pr exemplo a *Firewall*-1 da Checkpoint (Figura 3.3), um popular produto comercial na área, permite a selecção de quais os campos que se pretende registar no seu poderíamos seleccionar para registo no seu *log*.

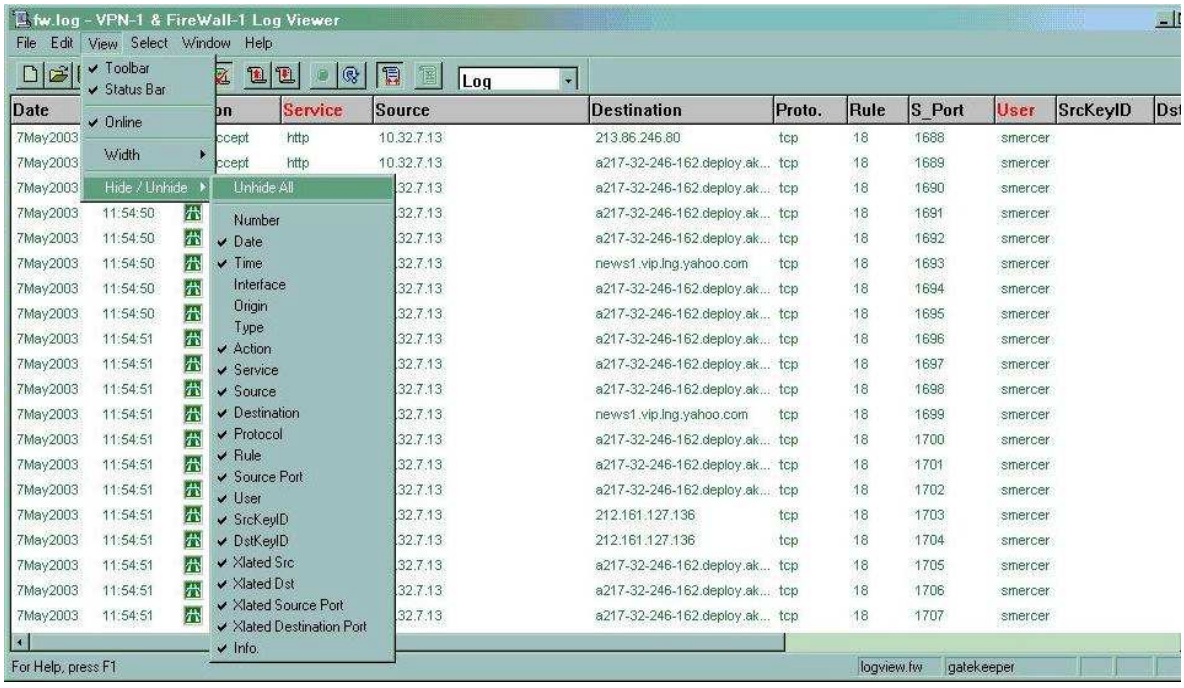

Figura 3.3 - Selecção dos campos na *Firewall*-1

No exemplo seguinte, podemos ver também o aspecto dos *logs* desta *Firewall* quando exportados para um ficheiro de texto. Todos os campos estão separados por espaços e o seu conteúdo está delimitado por aspas. Não seria, certamente, difícil a extracção dos dados relevantes.:

```
"4176"  "7May2003"  " 6:27:49"  "daemon"  "FW-1"  "log"  "accept"
"http"   "sukpc-emnt2.uk.ind.sonae"   "ad1.adsolution.de"   "tcp"
"18"  "1920"  "cbooth"  ""  ""  ""  ""  ""  ""  " reason previous
authentication                                          resource
http://as1.falkag.de/server/poor.asp?cmd=ban&kid=44065&dat=75243&rd
m=2003.05.07.14.27.32?"
"5028"  "7May2003"  " 7:59:51"  "daemon"  "FW-1"  "log"  "decrypt"
"lotus"    "host213-120-3-225.Webport.bt.net"    "liverpool-notes"
"tcp"  "17"  "1240"  "jhay"  "ac5b25eae2af"  "d888b97390dc"  ""  ""
""  ""  " scheme: FWZ methods: FWZ1,FWZ1,MD5"
```

3.6 Servidores Multimédia

Estes podem ser utilizados integrados com o servidor *Web* ou separadamente para libertar recursos dos servidores primários e visam complementar a oferta de conteúdos que o sítio *Web* disponibiliza. Como exemplo deste tipo de produtos temos o *Microsoft Media Services* ou o *Helix Universal Server* da *RealNetworks*. Servindo áudio ou vídeo a pedido, os seus *logs* serão úteis para perceber qual a reacção dos utilizadores ao conteúdo disponibilizado. Uma empresa que venda vídeos ou CD pode complementar o seu serviço disponibilizando anúncios sobre um DVD ou os primeiros segundos das músicas de um CD. Ao analisar os *logs* de acessos destes servidores multimedia podemos chegar à conclusão de que, por exemplo, uma determinada amostra de uma música é constantemente pedida. Este tipo de informação poderá ser um indicador para aumentar a publicidade e exposição do CD, ou dos autores, no sítio *Web*, por forma a canalizar o interesse dos utilizadores para a concretização de uma compra.

O servidor multimedia da *RealNetworks* tem capacidade de gerar 6 tipo de *logs* de acesso com um nível de detalhe variável [RealHelix03]. Neste documento apenas é analisado, a título de exemplo, um dos formatos de *log* disponibilizados (Tabela 3.11).

De seguida podemos ver um pequeno exemplo de um *log* de acessos gerado, com a configuração por omissão, pelo *Helix Universal Server*:

```
10.32.100.3 - - [26/Jun/2003:10:11:03 -0700] "GET showvideo.rm
RTSP/1.0" 200 858636
[WinNT_5.0_6.0.10.714_RealPlayer_RN92PD_en_686]
[8e07b707-19b7-448b-96b6-96c90151f2a6] [UNKNOWN] 826332 231 205 1 0
123
```

Campo	***Descrição***
Endereço IP do Cliente	Endereço IP do cliente que efectuou o pedido.
- -	Segunda e terceira posições não estão documentadas nos manuais.
Data e hora do acesso	Data e hora em que o cliente efectuou o pedido , segue o mesmo formato de data usado no *NCSA Common Log Format*.
Ficheiro e protocolo	Caminho e nome do ficheiro pedido bem como o protocolo utilizado. Se ficheiro não existe então o nome é `UNKNOWN`.
Bytes enviados	Número de *bytes* transferidos para o cliente.
Informação cliente	Descreve a versão e tipo do programa cliente usado.
Identificador do cliente	Identificador numérico único enviado pelo programa cliente, ou baseado no valor de uma *cookie*. Se este valor for desconhecido então este campo é preenchido com o valor [00000000-0000-0000-0000-000000000000].
Estatística cliente	Estatísticas sobre a conexão enviadas pelo cliente no final da exibição do ficheiro.
Tamanho do ficheiro	Tamanho do ficheiro em *bytes*.
Tempo de duração	Tempo de duração total do ficheiro em segundos.
Tempo de exibição	Tempo de exibição em segundos, poderá ser menor que a duração do ficheiro se a exibição foi interrompida antes do final.
Reenvios	Número de pacotes reenviados com sucesso devido a erros na transmissão.
Reenvios falhados	Número de pacotes reenviados mas que não chegaram a tempo de corrigir os erros gerados durante a transmissão.
Identificador da apresentação	Número atribuído pelo servidor a cada apresentação.

Tabela 3.11 – Campos registados no servidor multimedia da RealNetworks

3.7 Servidores *Web* Aplicacionais

As organizações poderão optar por utilizar programas que já implementam algum tipo de lógica e fluxos comerciais predefinidos, tais como uma loja virtual com sistema de catálogos, cesto de compras e registo de pagamentos, em vez de os desenvolver de raiz. Estes programas trabalham conjuntamente com os servidores *Web* e funcionam, geralmente, como uma ponte entre os servidores *Web* e as bases de dados que dão suporte ao negócio.

Embora o servidor *Web* seja capaz de efectuar a autenticação de utilizadores em ambientes transaccionais, esta função é normalmente efectuada a nível aplicacional. Serão, pois, estas aplicações *Web* que disporão dos mecanismos que efectuam o controlo e registo de acessos dos clientes. Esta informação é relevante para, por exemplo, efectuar monitorizações de acessos ao sítio *Web* por cliente. De igual forma, estas aplicações poderão ser o único ponto de registo das transacções efectuadas pela Internet sendo, portanto, relevantes para as análises de vendas por esse canal. Quando estamos a falar de aplicações *Web* distribuídas comercialmente, tais como *Microsoft Commerce Server*, *Blue Martini*, *ATG Dynamo*, *Intershop Infinity*, teremos de lidar com estruturas e sistemas de dados proprietários. Convirá que toda a estrutura seja conhecida e documentada. Caso contrário, estamos perante caixas negras que em nada contribuirão para a extracção dos dados necessários para o *Data Webhouse*.

3.8 Motores de Pesquisa

Nos pedidos HTTP referenciados pelos motores de pesquisa chegam palavras, ou frases, nos parâmetros do URI. Estas palavras, ou frases, são um óptimo indicador daquilo que os visitantes procuram. Pela análise destes dados, os conteúdos, ou *design*, do sítio *Web* podem ser mudados. Estas palavras podem também ser adicionadas às meta-etiquetas html que, normalmente, são consultadas pelos robots dos motores de pesquisa para efeitos de indexação das páginas *Web*. Desta forma, os motores de pesquisa orientarão os utilizadores mais rapidamente para as páginas que lhes possam interessar. É necessário também manter informação sobre como identificar os motores de pesquisa existentes para assim os poder distinguir de outros referenciadores com

menor interesse. Essa identificação poderá ser feita através do nome ou endereço IP que nos chega no URI. Para além de identificar quem são estes motores de pesquisa, deverá ser também identificado como é que eles efectuam a recolha de dados, ou seja, qual a identificação dos robots de recolha de dados que por eles são utilizados. Desta forma, poderemos correctamente identificar quais os acessos efectuados por estes robots que ficam registados nos *logs* do servidor *Web*.

3.9 Navegador e Computador do Visitante

O próprio navegador do cliente pode ser uma fonte de dados interessante. Poderão sempre existir navegadores construídos, ou alterados, para efectuar recolhas específicas de dados. Este método, no entanto, implica o consentimento do utilizador e apenas será expectável que um pequeno número de utilizadores o aceite. Poderá, eventualmente, ser um método aplicável em estudos, para efeitos de amostragem, em ambientes controlados e com um universo de utilizadores conhecido. Podemos, no entanto, e sem nenhuma alteração aos navegadores *standard* , obter um conjunto de informações interessantes através do uso de algumas instruções embebidas nas páginas *Web*: informação sobre a hora local do utilizador, resolução gráfica do écran, número de cores utilizado, língua configurada do navegador, se registo de *cookies* está bloqueado ou não, etc. Toda esta informação pode ser facilmente transmitida para o servidor *Web* através de, por exemplo, parâmetros passados no URI. Estes parâmetros podem ser processados pelo servidor *Web* para reconfigurar o ambiente e, mais relevante para o nosso caso, ficarem registados nos ficheiros de *log* e poderem ser posteriormente tratados no *Data Webhouse*.

As *cookies* podem ser geradas quer programaticamente, por código embebido nas páginas *Web*, quer automaticamente pelo servidor *Web*. Para o *Data Webhouse* o importante é que as *cookies* fiquem registadas nos *logs* do servidor *Web*. Apenas na resposta pelo servidor *Web* ao primeiro pedido HTTP efectuado é que elas podem ser atribuídas ao navegador do visitante. Isto quer dizer que o seu registo nos *logs* do servidor *Web* só será feito, quando activado, a partir do segundo pedido HTTP efectuado pelo navegador do visitante.

Um dos usos correntes das *cookies* é o rastreio de todos os pedido HTTP pertencentes a uma mesma visita, ou sessão, recorrendo a uma *cookie* transiente. Um outro uso comum, e se o seu

registo não estiver impedido no navegador do visitante, é a de utilização de uma *cookie* para identificar, por exemplo, visitas posteriores efectuadas a partir do mesmo navegador. Será de ter em conta que, com este método, só se poderá identificar o navegador e não obrigatoriamente o utilizador, já que o computador pode ser usado por mais do que uma pessoa.

3.10 Descritores de Estrutura e Conteúdo de Servidor *Web*

Nem sempre será possível inferir qual o conteúdo de uma página a partir do nome do ficheiro registado no *log* do servidor *Web*. As páginas *Web* deverão registos das associações entre os nomes das páginas e os seus conteúdos. Os programadores *Web* deverão manter numa base de dados o registo completo dos atributos dos objectos existentes no servidor *Web*, sejam eles páginas *Web*, imagens, ficheiros para descarga pelos visitantes ou qualquer outro elemento disponibilizado. Em [Sweiger et al. 02] e [KimballMerz00] são propostas e descritas duas estruturas para bases de dados que poderiam ser utilizadas para este efeito.

Elementos da área comercial ou *marketing* da empresa poderão também classificar e agrupar as páginas por áreas ou temas. Esta informação irá ajudar a compreender a navegação e objectivos das visitas ao servidor *Web*.

3.11 Logs de ISPs

Os *logs* dos *Internet Service Providers* (ISPs) registam toda a história de navegação de um utilizador. Estamos a falar essencialmente dos *logs* registados pelos seus *proxies*. Apenas os ISPs têm uma visão completa sobre os sítios e páginas que um utilizador visitou antes e depois do servidor *Web* em análise já que as suas *proxies* servem de intermediários entre o computador do utilizador e os vários servidores *Web* visitados. Para um sítio de cariz comercial poderá ser extremamente interessante saber que os seus utilizadores estão a visitar os sítios da concorrência depois de visitarem o seu. Isto poderá ajudar a desencadear medidas correctivas, sejam elas ao nível da estrutura do sítio ou dos conteúdos ou até mesmo dos preços. Esta informação, embora

possa ser útil, será, porventura, de difícil aquisição e limitada apenas ao universo de utilizadores servidos pelos ISPs considerados.

3.12 Dados de Servidores de Redes Publicicitárias

Se a organização publicita em outros sítios para atrair visitantes para o seu servidor *Web*, ou vende espaço publicitário nesse mesmo servidor, e utiliza redes externas de publicidade na Internet então os *logs* gerados por estas redes poderão complementar a informação de que dispomos com informações sobre o tipo de publicidade exibida e quantas selecções foram efectuados sobre os espaços publicitários. Com estes dados poderemos saber quais as campanhas e quais os sítios que mais visitantes atraem. No caso da venda de espaço publicitário, estes dados também ajudarão no cálculo dos valores a cobrar aos clientes desses espaços.

3.13 Sistemas Transaccionais de Suporte ao Negócio

Será de pressupor que as organizações que abrem uma presença na Internet já tenham um conjunto de sistemas operacionais, sejam eles sistemas de *Enterprise Resource Planning* (ERP), *Manufacturing Resource Planning* (MRP) ou outros, com informações sobre clientes, saldos de conta, pagamentos, créditos, informação de produtos e níveis de inventário, transacções comerciais através de outros canais de venda, etc. Nesta área existem diversos tipos de aplicações destinados tanto a empresas de grande dimensão, tais como os sistemas SAP R/3 [SAPR3] ou Oracle E-Business Suite [OracleEBS], como para as de dimensão mais reduzida, tais como o Primavera [Primavera] ou PHC [PHC].

Ora esta é a fonte principal dos *Data Warehouses* mais clássicos. Se o sítio, ou sítios, *Web* objectos de análise forem de cariz transaccional então será também aqui se irá buscar, certamente, grande parte dos dados necessários para efectuar no *Data Webhouse* as análises de índole mais comercial.

Com diferentes níveis de complexidade, versões ou fabricantes, este tipo de sistemas trabalha, no entanto, essencialmente sobre bases de dados relacionais. Será daí que se terá de extrair os dados

relevantes para o *Data Webhouse*. Esta é uma área já com algum nível de maturidade e existem no mercado diversos fornecedores de aplicações especificamente desenvolvidas para lidar com a extracção de dados destes sistemas. Estas aplicações de extração ora acedem directamente aos repositórios de dados ora tiram proveito de funcionalidades de importação e exportação disponíbilizados pelos sistemas operacionais

3.14 Sistemas de CRM e Gestão de Contactos

Para ter uma visão completa sobre todos os pontos de contacto com o cliente no *Data Webhouse*, deverão ser também recolhidos dados sobre os sistemas de gestão de contactos existentes na organização. Se, por exemplo, a organização dispuser de programas de gestão de centros de atendimento telefónico poder-se-á incluir um resumo deste ponto contacto através, não das gravações das chamadas telefónicas, mas sim da categorização dessas chamadas. Podemos, assim, por exemplo, contabilizar o número de chamadas que foram recebidas por cliente com o objectivo de saber o estado das suas encomendas ou o número de reclamações efectuadas.

Deverá também ser possível registar os contactos efectuados a partir da organização registando, por exemplo, a quem e quando foram enviadas brochuras publicitárias ou amostras. Este tipo de informação poderá ajudar a relacionar, por exemplo, a altura em que a publicidade foi emitida ou exibida e as compras efectuadas no sítio *Web* podendo assim medir-se o sucesso de uma determinada acção ou campanha de *marketing*. Para o registo deste tipo de dados estamos a falar de sistemas de CRM a funcionarem, sobretudo, sobre bases de dados relacionais.

3.15 Dados Demográficos e de Mercado

É sempre possível a aquisição de dados que nos permitam conhecer, por exemplo, hábitos de consumo em determinadas áreas ou evolução e tendências do mercado. Este tipo de dados poderá ajudar a medir o desempenho de sítios comerciais contra os seus concorrentes. Dados de organizações que se dedicam ao rasteio e construção de perfis de utilizadores na Internet poderão também ajudar a complementar a informação existente sobre os visitantes do sítio *Web*. Como

exemplo (Figura 3.4) podemos ver uma amostra de um relatório, distribuído comercialmente, sobre a distribuição de população por escalões etários para um conjunto de códigos postais [EmpowerGeog].

Age Rank Report
2003 Age Demographics

			Median	Population by Age							
Rank	**Name**	**Population**	**Age**	**0-13**	**14-24**	**25-34**	**35-44**	**45-54**	**55-64**	**65-74**	**75 Plus**
1	82001 Cheyenne	**34,259**	35.1	17.90%	16.20%	15.80%	15.20%	12.60%	8.50%	6.80%	6.90%
2	82601 Casper	**23,746**	35.4	18.10%	18.00%	13.30%	13.30%	14.50%	8.80%	6.30%	7.60%
	Subtotal of High	**58,006**	**35.2**	**18.00%**	**16.90%**	**14.80%**	**14.50%**	**13.40%**	**8.60%**	**6.60%**	**7.20%**
3	82443 Thermopolis	**4,625**	45.4	14.70%	14.40%	8.10%	12.20%	16.10%	13.60%	10.60%	10.30%
4	82514 Fort Washakie	**1,269**	28.2	25.50%	19.90%	13.00%	12.60%	12.10%	8.50%	4.60%	3.90%
5	82005 Fe Warren Afb	**1,054**	25	20.10%	29.80%	30.40%	16.10%	2.40%	0.70%	0.30%	0.30%
6	82433 Meeteetse	**885**	42.6	16.10%	12.50%	10.50%	14.70%	19.20%	12.60%	9.00%	5.40%
7	82442 Ten Sleep	**727**	44.4	16.50%	12.60%	7.80%	14.20%	18.00%	14.70%	10.20%	6.00%
	Subtotal of Average	**8,560**	**38.4**	**17.30%**	**16.70%**	**11.80%**	**13.20%**	**14.30%**	**11.20%**	**8.30%**	**7.30%**
8	82212 Fort Laramie	**294**	44.1	16.40%	14.10%	8.40%	12.50%	17.40%	13.80%	10.90%	6.50%
9	82512 Crowheart	**239**	42.6	17.60%	16.70%	7.90%	12.80%	17.60%	15.20%	6.40%	5.80%
10	82229 Shawnee	**123**	40.1	18.10%	17.00%	7.80%	15.40%	19.00%	12.40%	6.60%	3.60%
11	00075 Yellowstone Natl Park	**62**	39.3	25.80%	9.70%	8.10%	24.20%	19.40%	8.10%	4.80%	0.00%
	Subtotal of Below Average	**718**	**42.3**	**17.90%**	**15.10%**	**8.10%**	**14.10%**	**17.90%**	**13.50%**	**8.10%**	**5.20%**
	Grand Total	**67,283**	**35.7**	**17.90%**	**16.90%**	**14.30%**	**14.30%**	**13.60%**	**9.00%**	**6.80%**	**7.20%**

Figura 3.4 – Dados demográficos sobre distribuição populacional por escalões etários

Capítulo 4

Modelo Dimensional de um Data Webhouse

A modelação dimensional inerente a um *Data Webhouse* é em tudo idêntica à existente em outro qualquer projecto de *Data Warehouse* ou *Data Mart*. Em termos gerais, o objectivo é sempre o mesmo: definir um contexto válido, adequado e compreensível dentro do qual os factos a analisar façam sentido. Será função dos agentes de decisão de cada organização a escolha de quais são esses factos e de como eles devem ser entendidos e trabalhados. Deverão ser identificadas e associadas a cada um destes factos uma, ou mais, medidas que servirão de suporte ao tipo de análises que se pretende efectuar. Será deveras importante a escolha de qual o "grão" que cada tabela de factos deverá apresentar visto que é esta escolha que vai estabelecer qual o significado de cada registo e, consequentemente, qual o nível de detalhe possível em cada análise. Toda a modelação dimensional deverá começar pela escolha do grão pois é a partir dele que se poderá decidir quais os factos e dimensões que mais se lhe adequam. Como é óbvio, é aconselhável encontrar um ponto de equilíbrio entre aquilo que é possível realizar e aquilo que efectivamente trará benefícios palpáveis à organização. No domínio *Web*, as fontes de dados de *clickstream* podem ser verdadeiros gigantes de informação capazes de fazer crescer de forma astronómica qualquer tabela de factos. Logo, deve existir um certo cuidado na escolha do grau de detalhe, por forma a evitar um crescimento excessivo da tabela de factos que não apresenta um ganho significativo em termos de conhecimento.

Para a análise de dados de *clickstream* num *Webhouse* há, essencialmente, três tipos de grão que podem ser escolhidos:

- Um registo por cada ocorrência de um pedido HTTP a objectos individualmente endereçáveis no sítio *Web.*
- Um registo por cada pedido individual de páginas *Web* e respectiva actividade.
- Um registo por cada sessão, ou visita, completa.

4.1 Tabela de Factos para Análise de Pedidos http

No exemplo apresentado (Figura 4.1) podemos ver o esquema para uma tabela de factos desenhada com o objectivo de analisar os pedidos HTTP, também conhecidos como *hits*, registados num servidor *Web*. O tipo de análises conseguido nesta tabela terá maior uso essencialmente para administradores de servidores *Web*, designers e programadores, bem como gestores de conteúdo.

São doze as dimensões escolhidas, mais uma tabela auxiliar, para definir o contexto da ocorrência. Descrevemos, assim, quem pediu (dimensões `Utilizador`, `Computador do Utilizador`, `Entidade` e respectiva tabela auxiliar `Perfil da Entidade`), a quem e o quê (dimensões `Sítio Web`, `URI` e `Objecto Web`), quando isso aconteceu (dimensões `Data` e `Tempo`), de que forma (dimensões `Método HTTP` e `Agente http`), referenciado por quem (dimensão `Referenciador`) e qual foi o resultado do pedido (dimensão `Estado http`).

Analisando em pormenor o esquema apresentado, constata-se a existência de duas relações entre a tabela de factos e cada uma das dimensões `Data` e `Tempo`. Existem na tabela de factos duas chaves estrangeiras que identificam univocamente o momento relativamente a um padrão, sendo, neste caso, usado o valor GMT para a data e tempo. As outras duas chaves estrangeiras identificam quais eram os valores nesse mesmo momento que estavam localmente configurados no computador do utilizador, isto quando estes valores são possíveis de obter.

No mesmo esquema observa-se a existência de duas relações com a dimensão URI. Geralmente os pedidos por parte dos utilizadores são a páginas e não a objectos individuais. Estas duas relações visam então enquadrar os pedidos a URIs que identificam objectos, tais como imagens, e simultaneamente identificar a que página estes pertencem. Isto numa situação onde um URI de um objecto pertença apenas a uma página. Caso pertença a mais do que uma então o esquema teria de ser modificado para reflectir essa relação.

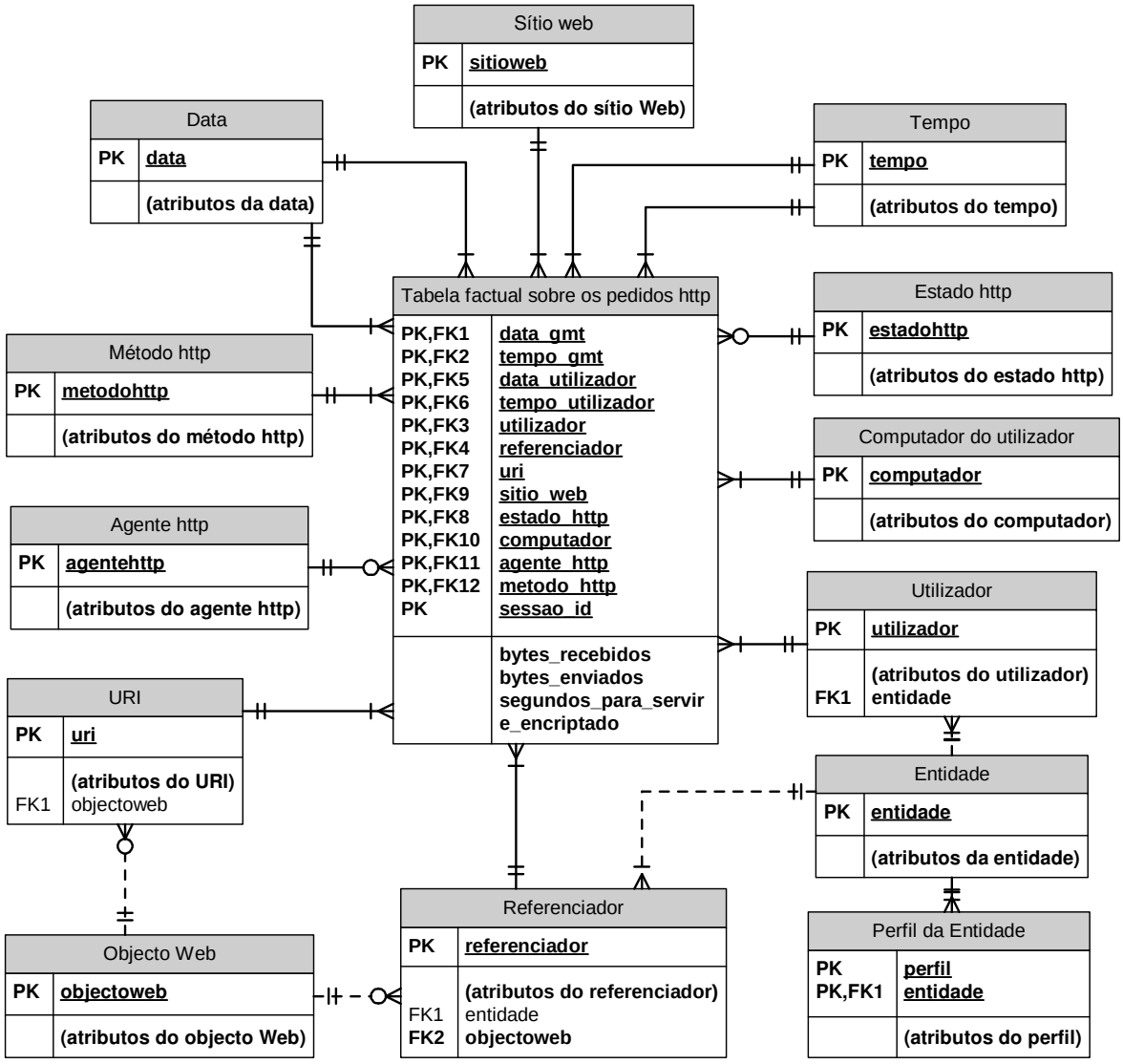

Figura 4.1 – Esquema dimensional para análise dos pedidos http

De todos os atributos que fazem parte da chave primária da tabela de factos é de notar a existência de um atributo sem dimensão associada: o identificador de sessão (`sessao_id`). Este valor visa apenas servir como elemento agregador dos registos pertencentes à mesma sessão, ou visita, do utilizador.

Os factos medidos são os seguintes:

- `Bytes recebidos` pelo servidor *Web*.
- `Bytes enviados` pelo servidor *Web*.
- `Segundos para servir` o pedido HTTP.
- Valor que indica se URI `É Encriptado`.

Como já se pode depreender pelos factos medidos, o grão desta tabela de factos corresponde a um registo por cada pedido HTTP individualmente recebido pelo servidor *Web*.
Com este *Data Mart* poderíamos obter respostas para, por exemplo, as seguintes questões:

- Que partes do sítio *Web* têm mais visitas?
- Em que páginas existem URLs que não levam a lado nenhum?
- Que partes do sítio *Web* são supérfluas ou visitadas menos frequentemente?
- Que páginas parecem ser aquelas onde mais visitantes terminam a sessão e abandonam o servidor *Web*?
- Quando é que o tráfego no sítio *Web* é mais elevado?
- Quando é que o tráfego no sítio *Web* é mais reduzido?
- Quantos *bytes* estão a ser transferidos?
- Qual o caminho de navegação mais popular? Podem ser melhoradas as páginas nesse caminho?
- Quem está a referenciar o nosso sítio *Web*?
- Se os visitantes utilizaram um motor de pesquisa para chegarem ao sítio *Web*, que palavras usaram eles na busca?
- Quais as páginas por onde os visitantes entram ou saem?
- Por quando tempo ficam os visitantes no sítio *Web*?
- Que navegadores *Web* usam os visitantes?
- Que motores de pesquisa estão a indexar o nosso sítio *Web*?

- Os empregados em escritórios remotos, ou parceiros de comerciais da empresa, estão a tirar partido dos recursos no sítio *Web?*
- Que tipos de organizações estão a visitar o sítio *Web*?
- Quantas visitas de utilizadores anónimos recebeu o sítio *Web*?
- Quantas visitas de utilizadores registados recebeu o sítio *Web*?
- Quantas visitas foram efectuadas por um utilizador anónimo antes de ele se registar e efectuar uma compra no sítio *Web*?
- Quem são os utilizadores que mais usam o sítio *Web*?
- Com que frequência os utilizadores registados visitam o sítio *Web*?
- Quando foi a última vez que visitaram o sítio *Web*?
- Quais são as partes do sítio *Web* mais visitadas por utilizadores registados?
- Qual o caminho de navegação mais comum para um utilizador anónimo?
- Qual o caminho de navegação mais comum para um utilizador registado?
- Qual a origem geográfica dos utilizadores?

4.1.1 Sítio *Web*

A dimensão `Sítio Web` (Tabela 4.1) visa primariamente identificar a que sítio *Web* foi efectuado o pedido HTTP. Esta poderá ser uma questão irrelevante mas, numa situação onde cada vez mais temos vários sítios *Web* alojados num único servidor *Web*, então será conveniente identificarmos devidamente o sítio da ocorrência. Esta dimensão assume que os vários sítios *Web* estão todos num único computador. No caso de termos o sítio, ou sítios, *Web* distribuídos por vários servidores, situação comum em fornecedores de serviços de alojamento que recorrem à utilização de *Web farms*, então haveria a necessidade de ter uma outra dimensão que identificasse qual o servidor que recebeu o pedido.

Atributo	***Descrição***
Chave Sítio Web	Valores Delegados.
Tipo	por exemplo, regular, desconhecido, corrupto.
Nome	Nome do sítio, ex: Sítio transaccional Empresa XPTY.
Nome DNS	Nome completo no domínio ex: www.someorg.biz.

Endereço IP	Endereço IP do sítio, ex: 10.32.100.3.
Descrição	Descrição textual do sítio.
Software Servidor Web	Nome e fabricante do software usado como servidor *Web*, ex: Microsoft IIS.
Versão Software	Versão do software, ex: Versão 5.
Nome computador	Nome principal do computador onde está alojado o sítio *Web*.
Nome do SO	Nome do sistema operativo do computador onde está alojado o sítio.
Versão SO	Versão do sistema operativo do computador onde está alojado o sítio.

Tabela 4.1 – Atributos da dimensão Sítio *Web*

4.1.2 Data e Tempo

De todas as dimensões referidas, as dimensões temporais, `Data` e `Tempo`, são talvez as mais óbvias e intuitivas, visto que um qualquer evento dificilmente fará sentido se estiver desprovido de uma referência desta natureza. Devido ao funcionamento do ambiente *Web*, é necessário manter um registo dos acontecimentos ao segundo. Se a informação fosse mantida numa única dimensão, seria necessário repetir a informação dos segundos para todos os dias. Isto aumentaria imenso o tamanho da dimensão optando-se assim pelo uso de duas dimensões distintas.

A dimensão Data (Tabela 4.2) reflecte o calendário, isto é, caracteriza o dia em que ocorreu o evento através do dia propriamente dito e do respectivo mês, semestre, ano e eventuais atributos "especiais" (por exemplo, a ocorrência de um feriado, um fim de semana, as férias, etc.). Por outro lado, a dimensão `Tempo` (Tabela 4.3) regista o momento do dia em que o evento ocorreu, permitindo diferentes níveis de indexação. Para além de manter os elementos no formato SQL da base de dados usada, identifica cada um destes separadamente, de modo a que seja possível analisar o que aconteceu num determinado intervalo de tempo.

Atributo	*Descrição*
Chave	Valores delegados.
Tipo de Data	por exemplo, regular, desconhecido, corrupto.
Formato SQL	Etiqueta temporalno formato da base de dados usada. Valor não nulo desde que o tipo seja regular.
Dia da semana	Segunda-feira,...,Domingo.
Nº do dia na semana	1,...,7.
Nº do dia no mês	1, 2,3, ..., 31.
Nº do dia no ano	1,2,3,...,365,366.
Tipo de dia	dia de trabalho, férias, ...
Feriado	nada ou o nome do feriado em causa.
Nº da semana no ano	1,...,53.
Mês	Janeiro, Fevereiro, ...
Nº de dias no mês	28,29,30,31.
Nº do mês no ano	1,...,12.
Nome do Trimestre	1Trim2003, 2Trim2003,...
Nº do Trimestre	1,2,3,4.
Nome do Semestre	por exemplo, 1S2003.
Nº do Semestre	1,2.
Ano	Expresso com quatro dígitos.
Nº de dias no Ano	365, 366.

Tabela 4.2 – Atributos da dimensão Data

No caso das análises que necessitarem de ser feitas abarcarem diferenças existentes a nível internacional então, no caso da dimensão `Data` (Tabela 4.2), atributos como a indicação de feriados, número de dia na semana e tipo de dia deveriam ser deslocados para uma tabela auxiliar que os associaria ao respectivo país. De igual forma, o atributo período na dimensão `Tempo` deveria ser também deslocado para uma tabela auxiliar com uma associação ao respectivo país.

Atributo	***Descrição***
Chave	Valores delegados.
Tipo de Tempo	Por exemplo: regular, desconhecido, corrupto.
Formato SQL	Etiqueta temporalno formato da base de dados usada. Valor não nulo desde que o tipo seja regular, por exemplo: 13:57:45.
Hora	0... 23.
Minuto	0...59.
Segundo	0...59.
Nº de segundos desde a meia-noite	0,...,89399.
Nº de minutos desde a meia-noite	0,...,1439.
Período	Manhã, Hora de Almoço, Tarde, Fim da tarde, Noite.

Tabela 4.3 – Atributos da dimensão Tempo

4.1.3 Método http

Esta dimensão (Tabela 4.4) indica qual foi o método usado pelo agente HTTP do utilizador para efectuar o pedido ao servidor *Web*. O conjunto de valores possíveis para esta dimensão é definido pelo protocolo HTTP [Fielding et al. 99][BernersLee et al. 96].

Atributo	***Descrição***
***Chave Método* HTTP**	Valores Delegados.
Tipo	Por exemplo: regular, desconhecido, corrupto.
Nome método	GET, POST, HEAD, PUT, DELETE, TRACE, CONNECT, OPTIONS.
Descrição	Descrição textual do método.

Tabela 4.4 – Atributos da dimensão Método http

A dimensão Método HTTP poderá ser utilizada, por exemplo, para auxiliar a identificação de tentativas de intrusão. Se o cliente HTTP fizer um pedido a uma página Web com um método que esta não suporta, ou diferente dos usuais GET e POST, então isto poderá indiciar um comportamento ilícito por parte do utilizador.

4.1.4 Agente http

O agente HTTP é o programa usado pelo utilizador para efectuar o pedido ao servidor. Esta informação é transmitida conjuntamente com o pedido efectuado ao servidor *Web* e permite-nos identificar, quando não dissimulada, o nome do navegador, a sua versão e em que sistema operativo corria quando efectuou o pedido. Na tabela apresentada (Tabela 4.5) podemos ver os atributos desta dimensão bem como uma breve descrição.

Atributo	***Descrição***
***Chave Agente* HTTP**	Valores Delegados.
Tipo	Por exemplo: navegador, agente automático, desconhecido, não classificado.
Texto identificador	Texto completo tal como registado no ficheiro de *log*.
Nome do Agente	Por exemplo: Microsoft Internet Explorer, Netscape Navigator, Opera.
Versão	Ex: 5.5, 4.7.
Sistema Operativo	Windows 98, Windows 95, HP-UX, AIX , MacOS.
Plataforma	Ex. Windows, Macintosh, Unix.

Tabela 4.5 – Atributos da dimensão Agente http

Ao conhecer os navegadores dos utilizadores, o sítio Web poderá eventualmente ser melhorado para tirar partido de uma ou outra funcionalidade que se sabe existir no navegador. Pode ser o caso da disponibilização de conteúdos adicionais, por exemplo animações multimédia, que se sabe que correm apenas em determinadas versões de navegadores.

4.1.5 Estado http

O estado HTTP é retornado pelo servidor *Web* e identifica qual foi o resultado do processamento do pedido. A dimensão `Estado HTTP` (Tabela 4.6) identifica e categoriza o significado desse estado.

Sabendo o estado retornado ao cliente HTTP podemos saber, por exemplo, quando estamos na presença de apontadores para páginas que não existem ou que já foram removidas. O código `404` seria o código retornado nesse caso .

Atributo	***Descrição***
***Chave Estado* HTTP**	Valores Delegados.
Tipo	Por exemplo: regular, desconhecido, corrupto.
Código do estado	Por exemplo: 200, 404, 500. Ver Anexo I - Códigos de estado http.
Descrição	Descrição textual segundo especificação do protocolo HTTP.
Categoria	2xx – Sucesso 3xx – Redireccionamento 4xx – Erro no cliente 5xx – Erro no Servidor

Tabela 4.6 – Atributos da dimensão Estado http

4.1.6 Computador do Utilizador

Na dimensão `Computador do Utilizador` (Tabela 4.7) identifica-se, com o melhor detalhe que for possível obter, o computador que foi usado pelo utilizador para efectuar o pedido HTTP ao servidor *Web*. Esta informação é reconstruída a partir do endereço IP ou nome DNS do computador.

A localização física do ponto de conexão do utilizador pode ser determinada a partir do endereço IP. Existem produtos e serviços que o conseguem fazer com elevado detalhe, podendo mesmo chegar à morada. Todavia, basta usar um simples utilitário tipo `whois`, disponível em vários sistemas operativos, para identificar o país da entidade detentora do endereço IP.

Atributo	*Descrição*
Chave Computador do Utilizador	Valores Delegados.
Tipo	Por exemplo: traduzido, não traduzido, desconhecido, corrupto.
Endereço IP	Endereço IP do computador, ou proxy, que efectuou o pedido HTTP.
Data da última alteração	Chave estrangeira para a dimensão Data. Regista a última vez que este registo foi alterado.
Nome DNS	Nome DNS completo traduzido a partir do endereço IP, por exemplo: pc53.norte.someorg.org ou desconhecido.
Nome do computador	Identificador, dentro do nome DNS, à esquerda do primeiro ponto contando da esquerda. Por exemplo: pc53 ou desconhecido
Rede DNS	Identificadores, dentro do nome DNS, à direita do primeiro ponto contando da esquerda. Por exemplo: norte.someorg.org ou desconhecido.
Domínio de topo	Por exemplo: org, com, biz, pt, uk, net ou desconhecido.
Primeiro nível do Domínio	Identificador do domínio de topo mais primeiro identificador à esquerda do primeiro ponto contando da direita . Por exemplo: sapo.pt, clix.pt ou desconhecido.
Segundo nível do Domínio	Identificador do domínio de topo mais primeiro identificador para a esquerda do primeiro ponto, mais primeiro identificador para a esquerda do segundo ponto contando da direita. Por exemplo: companhia.com.pt , norte.someorg.org, belo.nome.pt ou desconhecido.
Código ISO do País IP	Código ISO de duas letras da entidade detentora do endereço IP ou desconhecido.
Nome do País IP	Nome do país da entidade detentora do endereço IP ou desconhecido.

Tabela 4.7 – Atributos da dimensão Computador do Utilizador

4.1.7 Utilizador

Os atributos da dimensão `Utilizador` podem ser muito variados. A sua escolha dependerá imenso do objectivo do *Data Webhouse* e do tipo de análises que se pretendam fazer. A informação nesta dimensão poderá variar desde o simples registo de um *cookie* até à mais

completa informação demográfica. A dimensão utilizador aqui apresentada (Tabela 4.8) descreve vários atributos que poderão ser registados numa perspectiva de utilização do *Data Webhouse* para análise de um sítio *Web* transaccional.

Atributo	***Descrição***
Chave Utilizador	Valores Delegados.
Tipo	Por exemplo: cliente registado , cliente não registado, agente automático, desconhecido, corrupto
Identificador único	Atributo que identifique univocamente um utilizador.
Data da primeira visita	Chave estrangeira para a dimensão `Data`.
Hora da primeira visita	Chave estrangeira para dimensão `Tempo`.
Data da visita anterior	Chave estrangeira para a dimensão `Data`.
Hora da visita anterior	Chave estrangeira para dimensão `Tempo`.
Código do utilizador	Código de autenticação usado pelo utilizador , se utilizador registado, ou não utilizado.
Primeiro Nome	Primeiro nome declarado ou então toma o valor de Desconhecido.
Último Nome	Último nome declarado ou então toma o valor de Desconhecido.
Género	Masculino, Feminino, Não declarado, Desconhecido.
Endereço de email	Endereço de email declarado ou então toma o valor de Desconhecido.
Função	Função laboral desempenhada ou então toma o valor de Desconhecido.
Data da última compra	Chave estrangeira para a dimensão `Data`.
Número de telefone	Número de telefone declarado ou então toma o valor de Desconhecido.
Número de telemóvel	Número de telemóvel declarado ou então toma o valor de Desconhecido.
Entidade	Chave estrangeira para dimensão Entidade que identifica detalhes adicionais, por exemplo de moradas, do indivíduo ou organização visitantes.

Tabela 4.8 – Atributos da dimensão Utilizador

4.1.8 Entidade e Perfil de Entidade

A dimensão `Entidade` (Tabela 4.9) agrupa informação sobre indivíduos ou organização. Será uma forma de, por exemplo, identificar as visitas de vários utilizadores pertencentes a uma mesma organização, informação bastante importante quando estamos a falar de relações comerciais com outras organizações.

Atributo	*Descrição*
Chave Entidade	Valores delegados.
Tipo de Entidade	Por exemplo: indivíduo, organização, desconhecido, corrupto, inaplicável.
Nome da entidade	Nome da organização ou nulo.
Número de identificação fiscal	Número de contribuinte.
Nome do contacto principal	Por exemplo: José Silva.
Número de telefone principal	Por exemplo: +351 123456789.
Número de fax principal	Por exemplo: +351 123456780.
Email do contacto principal	Por exemplo: jsilva@hotmail.com.
URL	Por exemplo: http://www.someorg.org.
Tipo de morada	Por exemplo: morada de entrega, morada de facturação, morada de comprador.
Nome da Rua	Por exemplo: Rua de Gualtar.
Número	Por exemplo: 99.
Localidade	Por exemplo: Gualtar.
Cidade	Por exemplo: Braga
Região	Por exemplo: Minho.
País	Por exemplo: Portugal

Tabela 4.9 – Atributos da dimensão Entidade

Os dados incluídos nesta dimensão tentam ser genéricos. Informação comercial específica terá de ser acrescentada em tabelas auxiliares associadas a esta dimensão. Será o caso de, por exemplo, querermos manter informações de todas as moradas de uma entidade para análises por morada de entrega.

Na dimensão Entidade poderemos ter, por exemplo, dados sobre um motor de pesquisa que pode ser, num momento, a entidade detentora do robot de indexação que visita o sítio *Web* e noutro momento ser a entidade referenciadora. Poderá acontecer o caso de existir um cliente que também é um fornecedor. A tabela `Perfil de Entidade` (Tabela 4.10) funciona como tabela auxiliar que visa identificar todos os papéis que uma dada entidade toma num determinado momento no tempo.

Atributo	***Descrição***
Chave Perfil de Entidade	Valores delegados.
Tipo de perfil	Por exemplo: regular, desconhecido, corrupto.
Entidade	Chave estrangeira para dimensão Entidade
Descrição	Por exemplo: cliente, fornecedor, motor de pesquisa, referenciador, organização interna.
Data início	Chave estrangeira para dimensão `Data`. Indica quando é que a Entidade passou a ter este perfil
Data fim	Chave estrangeira para dimensão `Data`. Indica quando é que a Entidade deixou de ter este perfil.

Tabela 4.10 – Atributos da tabela auxiliar Perfil de Entidade

4.1.9 Referenciador

A dimensão `Referenciador` (Tabela 4.11) contém os atributos que identificam qual foi o URI que referenciou o pedido HTTP. Este URI tanto pode ser interno como externo ao sítio *Web* em análise. No caso de ser um referenciador externo, os dados poderão ajudar a validar o sucesso ou insucesso de qualquer campanha promocional em sítios *Web* externos.

Atributo	***Descrição***
Chave Referenciador	Valores delegados.
Tipo de referenciador	Por exemplo: interno ao sitio *Web*, externo ao sítio *Web*, não especificado, inaplicável, corrupto.
URI Referenciador	Por exemplo: http://www.someorg.net/links.html.
É motor de pesquisa	Sim ou Não.
Parâmetros do URI	Quando o referenciador é um motor de pesquisa este campo contém a expressão pesquisada pelo utilizador nesse motor.
Endereço IP referenciador	Endereço IP do computador que referenciou.
Data da última alteração	Chave estrangeira para a dimensão Data. Regista a última vez que este registo foi alterado.
Porta IP do referenciador	Por exemplo 80 ou 443.
Nome DNS completo do referenciador	Nome DNS completo traduzido a partir do endereço IP. Por exemplo: www.divisision.someorg.net ou desconhecido.
Nome do computador referenciador	Identificador, dentro do nome DNS, à esquerda do primeiro ponto contando da esquerda. Por exemplo: www ou desconhecido.
Rede DNS do referenciador	Identificadores, dentro do nome DNS, à direita do primeiro ponto contando da esquerda. Por exemplo: division.someorg.org ou desconhecido.
Domínio de topo do referenciador	Por exemplo: org, com, biz, pt, uk, net ou desconhecido.
Primeiro nível do Domínio do referenciador	Identificador do domínio de topo mais primeiro identificador à esquerda do primeiro ponto contando da direita. Por exemplo: sapo.pt, clix.pt ou desconhecido.
Segundo nível do Domínio do referenciador	Identificador do domínio de topo mais primeiro identificador para a esquerda do primeiro ponto, mais primeiro identificador para a esquerda do segundo ponto contando da direita. Por exemplo: companhia.com.pt , norte.someorg.org, belo.nome.pt ou desconhecido.
Código ISO do País IP do referenciador	Código ISO de duas letras da entidade detentora do endereço IP ou desconhecido.
Nome do País IP do	Nome do país da entidade detentora do endereço IP ou desconhecido.

referenciador	
Entidade	Chave estrangeira para dimensão Entidade que identifica detalhes adicionais, por exemplo de moradas, do indivíduo ou organização referenciador.
Objecto Web	Chave estrangeira para a dimensão Objecto *Web* que indica qual o objecto *Web* referenciador do pedido, por exemplo uma página *Web* interna que referencia uma imagem. Para referenciadores externos o Objecto *Web* indicado por esta chave tomará sempre o valor de "desconhecido".

Tabela 4.11 – Atributos da dimensão Referenciador

A dimensão `Referenciador` tem certas semelhanças com a dimensão `Computador do Utilizador`. Aqui poder-se-ia pensar numa normalização dos dados mas a probabilidade de termos, neste caso, dados duplicados não será elevada. A natureza dos mesmos tende a ser diferente já que um computador referenciador tenderá a ser um servidor *Web* HTTP enquanto que um computador do utilizador tenderá a ser um cliente HTTP.

4.1.10 URI

Esta dimensão (Tabela 4.12) contém o URI que foi pedido ao sítio *Web*, com exclusão da componente identificativa do sítio *Web* e das portas IP. Poder-se-ia pensar que a dimensão `Objecto Web` poderia desempenhar o mesmo efeito, mas tal não é verdade pois um mesmo objecto *Web* poderá ser invocado com diferentes parâmetros. Em situações de geração dinâmica de páginas *Web*, por exemplo, tendo um programa que apresenta as páginas em função de um identificador da página passado como parâmetro, esta dimensão ajudará a identificar qual, ou quais, foram as páginas servidas ao utilizador.

No URI podem ser passadas uma ou mais variáreis. Se houver intenção de analisar individualmente cada variável então podemos usar uma tabela auxiliar onde o nome e conteúdo dessas variáveis são colocados separadamente.

Em situações de sítios *Web* com URI gerados totalmente de forma dinâmica podemos não ter sequer a existência de objectos físicos em disco. Neste caso, a chave estrangeira para o objecto *Web* poderá apontar simplesmente para o registo que indica um objecto *Web* do tipo

"Desconhecido". Teremos uma situação similar para pedidos a objectos que nem sequer existam no sítio *Web*.

Atributo	***Descrição***
Chave URI	Valores delegados.
Tipo de URI	Por exemplo: regular, inaplicável, desconhecido, corrupto.
URI	Valor registado sem a componente identificadora do sítio *Web* e portas IP. Ex: `/produtos/catalogo.asp?id=1234&cat=escritorio`
Parâmetros do URI	Todo o texto à direita do símbolo '?' (ponto de interrogação) até ao final do URI ou até ao símbolo '#' (cardinal), para o caso ilustrado seria: `id=1234&cat=escritorio`.
Objecto Web	Chave estrangeira que identifica qual o objecto *Web* que o URI identifica.

Tabela 4.12 – Atributos da dimensão URI

4.1.11 Objecto *Web*

Qualquer objecto presente no sítio *Web* deverá estar descrito na dimensão `Objecto Web` (Tabela 4.13). Estamos a falar de ficheiros de imagens, ficheiros de som ou vídeo, ficheiros `html`, `xml`, `asp`, `aspx`, `php`, etc. Cada objecto deverá estar categorizado e descrito pela sua função principal. O valor e a diversidade destes elementos são determinados pela forma como os designers, programadores e elementos das equipas de *marketing* os caracterizam. Não esquecer, contudo, que o modelo de classificação e caracterização deverá ser flexível o suficiente para considerar a evolução do sítio *Web* ao longo do tempo.

Atributo	***Descrição***
Chave Objecto	Valores delegados.
Tipo de Objecto	Por exemplo: página, imagem, áudio, vídeo, desconhecido, corrupto.
Sítio Web do objecto	Chave estrangeira para a dimensão Sítio *Web* que identifica a que sítio este objecto pertence.
Directório	Caminho completo que identifica univocamente o directório de localização do objecto dentro do sítio *Web* a partir da sua raiz. Por exemplo: `/imagens/`, `/produtos/`.

Nome do ficheiro	Por exemplo: `index.asp`, `logo.gif`.
Tipo MIME	Por exemplo: `text/html`, `text/plain`, `text/xml`, `application/pdf`, `vídeo/x-mng`, `image/jpeg`, `image/gif`, `application/x-shockwave-flash`.
Nome do objecto	Nome do objecto, caso este o tenha. Em páginas *Web* será o nome que aparecerá na etiqueta html `<title>`.
Descrição do Objecto	Descrição textual da função principal do objecto.
Categoria do Objecto	Por exemplo: Página Raiz, Navegação, Conteúdo, Híbrido, Aplicação CGI, Ornamental, Desconhecido, Não aplicável.
Língua do Objecto	Língua em que o objecto está expresso. Por exemplo: Português, Inglês, Desconhecido ou Não Aplicável.
Tamanho em bytes do objecto	Por exemplo: 124
Versão do objecto	Por exemplo: 1, 3.1.
Data de criação	Chave estrangeira para a dimensão `Data`. Regista o dia em que o objecto foi criado.
Data da última alteração	Chave estrangeira para a dimensão `Data`. Regista o dia em que o objecto foi alterado pela última vez.
Data de remoção	Chave estrangeira para a dimensão `Data`. Regista o dia em que o objecto foi removido do sítio *Web*

Tabela 4.13 – Atributos da dimensão Objecto *Web*

4.2 Tabela de Factos para Análise de Utilização de Páginas *Web*

No exemplo ilustrado (Figura 4.2) podemos ver o esquema para uma tabela de factos desenhado com o objectivo de analisar os pedidos de páginas *Web* bem como a actividade relativa ao cesto de compras nessas páginas. De certa forma, esta tabela é o resultado da agregação da tabela de factos para análise de pedidos HTTP. Nessa agregação são eliminados todos os pedidos a objectos que não sejam páginas *Web* e o grão da tabela de factos passa a corresponder a um registo por cada página *Web* servida pelo sítio *Web*. O esquema apresentado é apenas um exemplo de um

modelo dimensional que poderia ser utilizado num contexto de um sítio *Web* transaccional. Se não existirem transacções comerciais então serão várias as modificações possíveis. Por exemplo, deixa de fazer sentido a existência de uma dimensão `Cesto de Compras`.

Ao associar os pedidos de páginas e a actividade relativa ao cesto de compras é possível relacionar directamente navegação e comportamento dos utilizadores com actividades de cariz comercial. Provavelmente serão os designers e programadores do sítio *Web*, bem como elementos ligados à área comercial e *marketing*, que maior partido tirarão das interrogações possíveis com este esquema.

Para além de ainda conseguir responder a muitas das perguntas que a tabela de factos para análise de pedidos HTTP responde, podemos ainda obter respostas para, por exemplo, as seguintes questões:

- Em que páginas os utilizadores ficam mais tempo?
- Quais as páginas que mais tempo demoram a ser servidas?
- Quais os produtos, ou categoria de produtos, mais vendidos através do sítio *Web*?
- Quais os produtos que mais vezes são adicionados e removidos do cesto de compras?
- Quanto é o tempo médio para efectuar uma compra?
- Quantos são os utilizadores que iniciaram uma compra e a completaram e quantos a abandonaram antes de a submeter?
- Em que ponto uma transacção comercial foi abandonada?
- Qual o valor perdido nos cestos de compras em transacções abandonadas?
- Existe alguma consistência no valor dos cestos de compras abandonados?
- Quais são as promoções *on-line* que maior volume de negócios, ou volume de visitas, atraem?

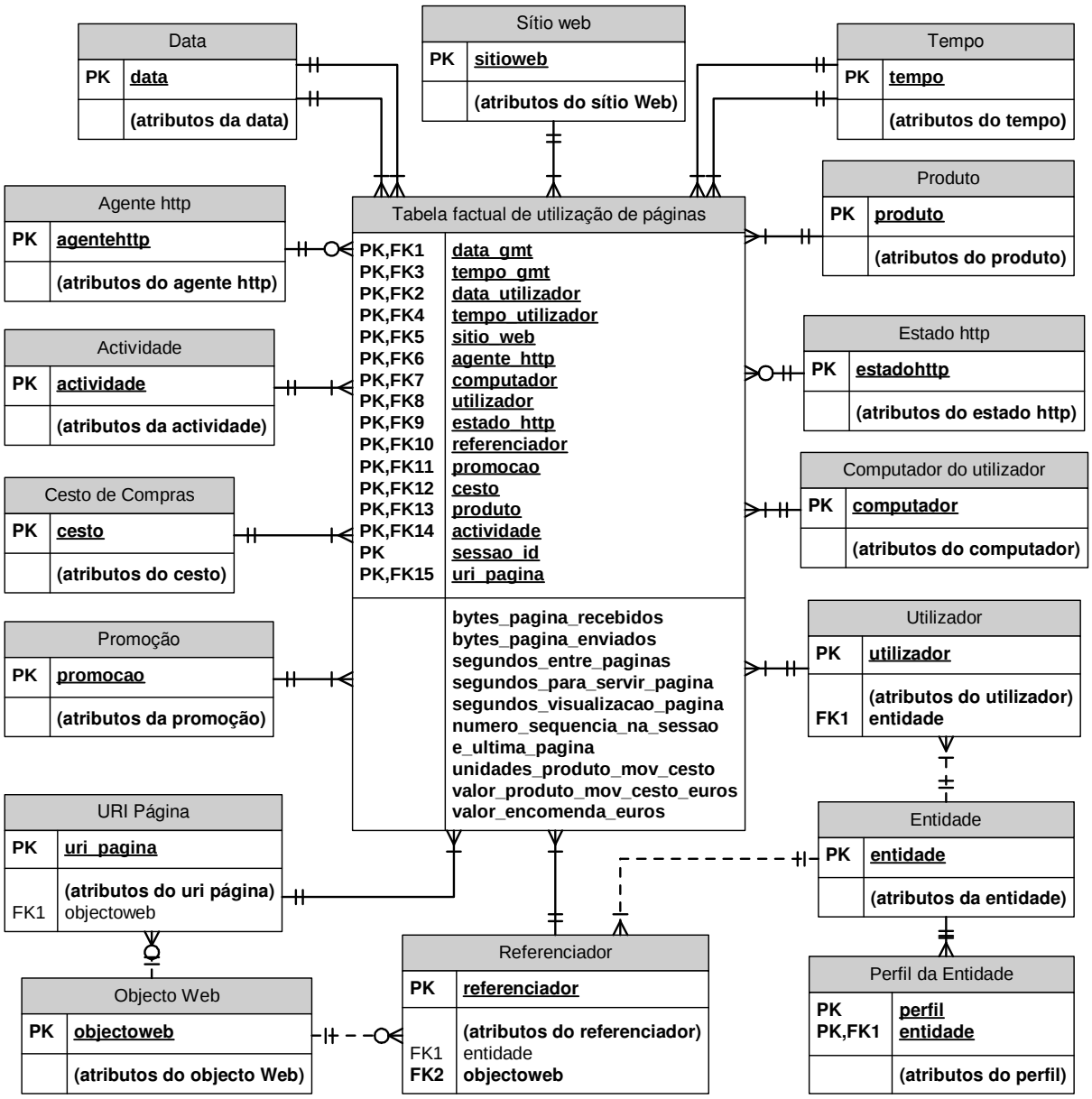

Figura 4.2 – Esquema dimensional para pedidos de páginas

Os factos medidos são os seguintes:

- `Bytes Página Recebidos`: o número de *bytes* recebidos pelo sítio *Web* para efectuar o pedido da página completa ao servidor.

- Bytes Página Enviados: o número total de *bytes* enviados pelo sítio *Web* ao navegador do utilizador.
- Segundos Entre Páginas: o terceiro facto é intervalo de tempo em segundos entre o pedido de uma página e o pedido da página seguinte.
- Segundos Para Servir Página: Número total de segundos que uma página demorou a ser servida incluindo o carregamento dos objectos que dela fazem parte.
- Segundos Visualização Página: Tempo estimado que a página esteve visível no navegador do utilizador.
- Número Sequência na Sessão: Valor numérico que indica, no contexto de uma sessão, em que posição foi a página servida ao utilizador.
- É Última Página: medida que toma o valor de **Sim** ou **Não** e que indica se uma determinada página foi a última página servida ao utilizador no contexto de uma sessão.
- Unidades Produto Movimentadas para/de Cesto: número de unidades do produto que são adicionadas, ou removidas, ao cesto de compras.
- Valor Produto Movimentado para/de Cesto em Euros: Valor do produto adicionado, ou removido, ao cesto de compras.
- Valor da Encomenda em Euros: Valor numérico que representa o valor em Euros de uma compra quando a actividade registada na página for Submeter Encomenda. Em todos os outros registos tomará o valor zero.

São quinze as dimensões escolhidas, mais uma tabela auxiliar, para definir o contexto da utilização da página: o Sítio *Web*, a Data de calendário, o Tempo do dia, o Agente HTTP, o Computador do Utilizador, a Promoção, o Produto, o Cesto de Compras, a Actividade, o Estado HTTP, o Utilizador, a Entidade e a respectiva tabela auxiliar Perfil da Entidade, o Referenciador, e finalmente o URI da Página e o Objecto *Web*.

As dimensões Sítio *Web*, a Data de calendário, o Tempo do dia, o Agente HTTP, o Computador do Utilizador, o Estado HTTP, o Utilizador, a Entidade e a respectiva tabela auxiliar Perfil da Entidade, o Referenciador e o Objecto *Web* são iguais às

associadas à tabela de factos de pedidos HTTP. Para esta tabela de factos as novas dimensões são o `Produto`, `URI Página`, `Promoção`, `Cesto de Compras` e `Actividade`.

4.2.1 Produto

A dimensão `Produto` (Tabela 4.14) será uma dimensão que apenas fará sentido em *Webhouses* para análise de sítios *Web* transaccionais. Se esse não for o caso, esta dimensão poderá ser totalmente descartada. Neste caso, a opção foi a de apresentar uma dimensão que representasse os produtos comercializáveis. Contudo, a existência de uma dimensão `Serviços` pode ser também considerada em complemento, ou em substituição, da dimensão `Produto`. As variantes possíveis para a dimensão `Produto` ou `Serviços` são enormes, sendo que a sua estrutura deve ser ajustada ao propósito específico de cada sítio.

Atributo	***Descrição***
Chave Produto	Valores delegados.
Tipo	Por exemplo: regular, desconhecido, corrupto.
Código Operational	Código interno usado nos sistemas operacionais.
Código EAN	Código único atribuído pela *European Article Numbering Association*.
Descrição	Descrição textual do produto.
Categoria	Categoria ou família de produtos. Poderão existir vários atributos semelhantes em função das necessidades de análise.
Fornecedor	Identificador de fábrica interna ou fornecedor externo. Poderá ser um campo com chave estrangeira para dimensão Entidade.
Unidade de venda	Por exemplo: M2, M3, Unidade, Caixa de 100.
Volume de embalagem	Valor numérico expresso em M3.
Peso de embalagem	Valor numérico expresso em Kilogramas.

Tabela 4.14 – Atributos da dimensão Produto

A diversidade dos atributos da dimensão `Produto` pode ser imensa. Podemos ir do simples código e descrição até à complexidade de uma representação de um bem comercializável apenas expresso por um conjunto de características ou uma receita de produção. Embora aqui apenas tenha sido incluído um atributo `Categoria`, a existência de vários níveis de classificação poderá ser

considerada por forma a poder constituir vários níveis hierárquicos que satisfaçam os requisitos de análise da organização.

4.2.2 URI Página

A dimensão `URI Página` (Tabela 4.15) poderá ou não existir. Esta resulta de uma refinação da dimensão `URI` onde todos os URIs não considerados como páginas *Web*, por exemplo de pedidos a ficheiros de imagens, são simplesmente excluídos. Se no *Webhouse* coexistirem as tabela de factos para análise dos pedidos HTTP e a tabela de factos para análise de utilização de páginas *Web* poderíamos manter unicamente a dimensão `URI`.

Tal como no caso da dimensão `URI`, também a dimensão `URI Página` será complementada com informações constante na dimensão `Objecto Web`.

Atributo	***Descrição***
Chave URI Página	Valores delegados.
Tipo de URI	Por exemplo: regular, inaplicável, desconhecido, corrupto.
URI	Valor registado sem a componente identificadora do sítio *Web* e portas IP. Ex: `/produtos/catalogo.asp?id=1234&cat=escritorio`
Parâmetros do URI	Todo o texto à direita do símbolo '?' (ponto de interrogação) até ao final do URI ou até ao símbolo '#' (cardinal), para o caso ilustrado seria: `id=1234&cat=escritorio`.
Objecto Web	Chave estrangeira que identifica qual o objecto *Web* que o URI identifica, mesmo que seja do tipo desconhecido.

Tabela 4.15 – Atributos da dimensão URI Página

4.2.3 Promoção

Um registo na dimensão `Promoção` (Tabela 4.16) identifica a localização e o tipo de promoções *on-line* efectuadas a um produto ou serviço. Estas promoções tanto podem ser internas como externas ao sítio *Web*. Ao comparar a informação contida nesta dimensão com a informação contida na dimensão Referenciador podemos aferir directamente o sucesso, ou insucesso, que uma

dada promoção está a ter no incremento de visitas ao sítio *Web*. Conjugando a informação das dimensões `Promoção`, `Referenciador` e esta informação sobre compras, podemos também medir o sucesso de uma dada promoção no nível de incremento das vendas.

Atributo	***Descrição***
Chave Promoção	Valores delegados.
Tipo de Promoção	Por exemplo: interna, externa, desconhecido, corrupto.
Sítio Web promovido	Chave estrangeira para a dimensão Sítio *Web* que identifica a que sítio esta promoção se refere.
Nome da promoção	Por exemplo: Saldos de Inverno.
Descrição da promoção	Descrição textual.
URI do item promotor	URI que identifica a localização do item publicitário, por exemplo: `http://www.sitiopromotor.org/banners/pub.html`.
Método de promoção	Por exemplo: banner, apontador, desconhecido.
Produto promovido	Chave estrangeira para a dimensão Produto que identifica qual o produto que está a ser promovido.
URI destino	URI que identifica para onde se será redireccionado caso se clique sobre o item publicitário.
Data de inicio	Chave estrangeira para a dimensão Data. Regista o dia em que a promoção se iniciou.
Data da última alteração	Chave estrangeira para a dimensão Data. Regista o dia em que a promoção foi alterada pela última vez.
Data do fim	Chave estrangeira para a dimensão Data. Regista o dia em que a promoção foi removida do sítio *Web.*

Tabela 4.16 – Atributos da dimensão Promoção

Em [KimballMerz00] é sugerida a existência de uma dimensão genérica onde se identifiquem causas ou circunstâncias externas que possam influenciar o número de visitas ao sítio *Web*. Seja, por exemplo, para identificar referências em serviços noticiosos, participações em feiras ou exposições, actos ou eventos resultantes de acção humana ou da natureza tipo inundações, tremor de terras, actos terroristas, etc.

4.2.4 Cesto de Compras e Actividade

Uma dimensão que represente o cesto de compras (Tabelas 4.17) apenas faz sentido existir num ambiente comercial. Cada registo nesta dimensão identifica univocamente um cesto de compras, dentro de um determinado sítio *Web*. A existência da dimensão `Actividade` (Tabela 4.18) visa, neste caso, a identificação de actividades registadas na página *Web* que sejam relevantes para o cesto de compras:

- Adicionar ao cesto.
- Remover do Cesto.
- Submeter Encomenda.

Outras actividades podiam ser identificadas mas estas serão, contudo, as que maior significado terão no contexto de um sítio *Web* transaccional.

Atributo	***Descrição***
Chave Cesto de Compras	Valores delegados.
Tipo de Cesto	Por exemplo: regular, desconhecido, corrupto.
Sítio Web	Chave estrangeira para a dimensão Sítio *Web* que identifica a que sítio este cesto pertence.
Identificador do Cesto	Código que identifique univocamente o cesto de compras no ambiente operacional.

Tabela 4.17 – Atributos da dimensão Cesto de Compras

Atributo	***Descrição***
Chave Actividade	Valores delegados.
Tipo de Actividade	Por exemplo: regular, desconhecido, corrupto.
Descrição	Por exemplo: Adicionar ao Cesto, Remover do Cesto, Submeter encomenda.

Tabela 4.18 – Atributos da dimensão Actividade

A informação fornecida por estas duas dimensões conjuntamente com os factos medidos da quantidade e valor dos produtos adicionados, ou removidos, poderão dar informações sobre a variação do valor do cesto e relacioná-lo directamente com a navegação do utilizador no sítio *Web*.

Se os utilizadores finais do *Data Webhouse* pretenderem aprofundar o estudo da actividade registada no cesto de compras então poder-se-á fazer evoluir a dimensão `Cesto de Compras` para uma tabela de factos optimizada para o efeito. Informações como, por exemplo, quais os produtos, ou serviços, que estariam no cesto em qualquer momento poderão ser obtidas com interrogações mais eficientes tendo uma dimensão `Linhas de Cesto` associada a essa nova tabela de factos.

4.3 Tabela de Factos para Análise de Sessões Completas

As análises possíveis ao nível do pedido HTTP ou das páginas *Web* poderão não justificar as necessidades de armazenamento que advém desse grão. Por outro lado, se os utilizadores finais necessitarem de dados que permitam ter uma visão mais abrangente sobre a evolução do sítio *Web* e resultados comerciais resultantes então o grão dessas tabelas não será o mais indicado. Pode-se então agrupar todos os pedidos HTTP por forma a permitir análises sumárias ao nível da sessão, ou visita. Nesse caso, existirá uma tabela de factos cujo grão corresponde a um registo por cada sessão completa de um visitante.

Povemos ver no exemplo ilustrado (Figura 4.3) um possível esquema dimensional para uma tabela de factos para análise de sessões completas. O esquema é bastante semelhante ao esquema dimensional apresentado para análise de utilização de páginas *Web*. Não são acrescentadas novas dimensões mas, pelo contrário, dimensões cujo grão não se adequa ao grão especificado para esta tabela de factos foram removidas. Por exemplo, a existência de uma dimensão `Produtos` associada a esta tabela de factos não faria sentido já que, numa sessão, podem ser manipulados vários produtos. Numa primeira análise, alguns poderão pensar que a dimensão `URI Página` também é inadequada. Todavia, esta dimensão está presente no esquema não para identificar todos os URIs de páginas que o utilizador percorreu mas sim com o objectivo de identificar dois pontos importantes no contexto de uma sessão: o URI da página de entrada e o URI da página de

saída. De igual forma, a dimensão `Referenciador` passou apenas a identificar quem foi o referenciador inicial da sessão do utilizador.

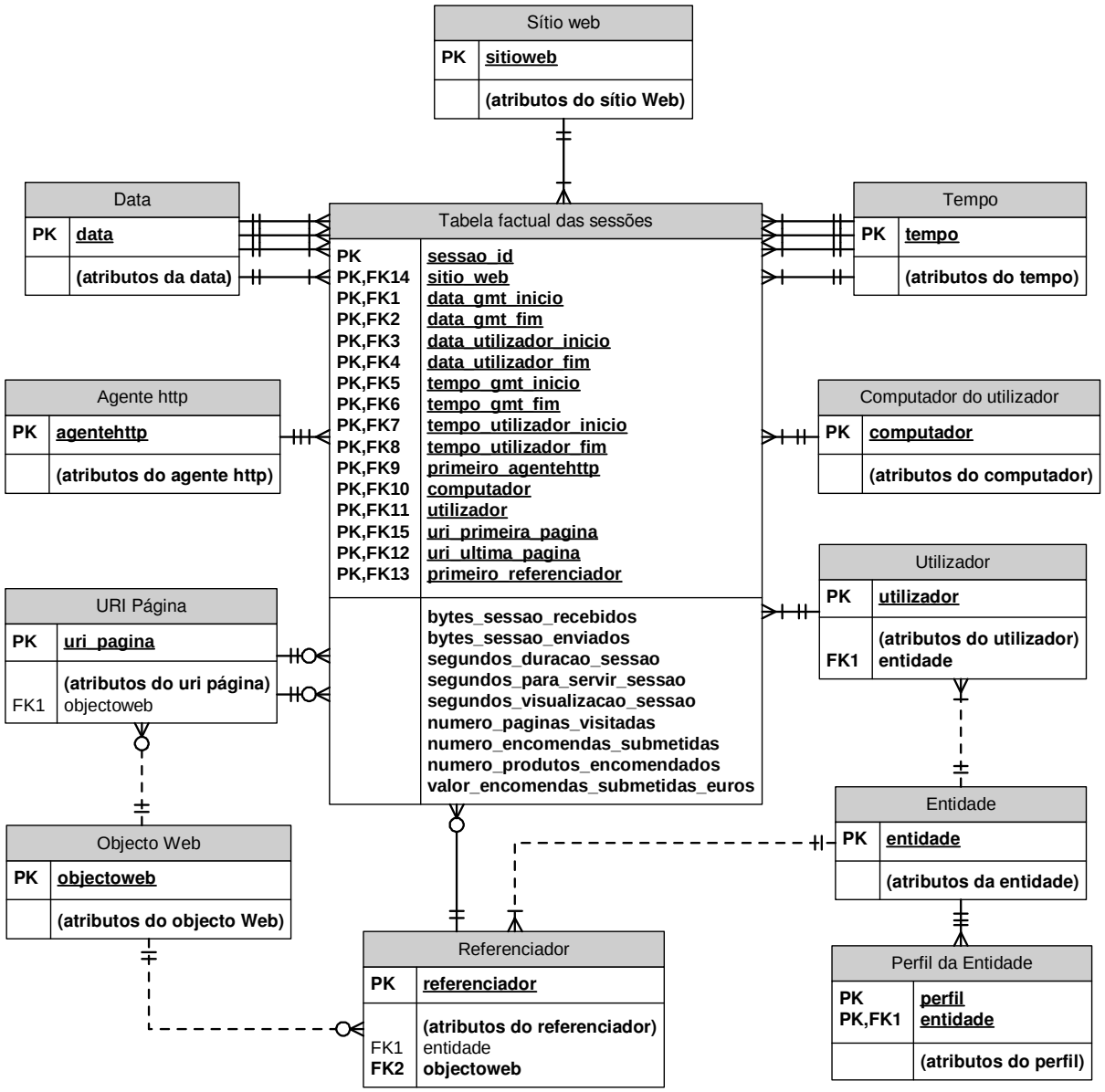

Figura 4.3 - Esquema dimensional para análise de sessões completas

Com uma tabela de factos para análise de sessões completas poder-se-á responder mais eficientemente a questões como:

- Quem são os melhores referenciadores externos, em número de visitas e em volume de negócio?
- Quanto tempo os utilizadores permanecem no sítio *Web* e quantas páginas visitam?
- Qual a percentagem de sessões que resultam em vendas?
- Qual o volume de vendas conseguido através de um sítio *Web* para um determinado período de tempo?
- Quais as áreas de entrega para vendas efectuadas através do sítio *Web*?

De uma forma geral, com este esquema, é possível responder a questões relacionadas com a identidade do utilizador, qual a página de entrada no sítio *Web*, por onde saiu e qual foi o resultado da sessão.

Os factos medidos nesta tabela são os seguintes:

- `Bytes Sessão Recebidos`: Este é o número de *bytes* recebidos pelo sítio *Web* durante a sessão.
- `Bytes Sessão Enviados`: O valor neste caso é o número de *bytes* enviados pelo Sítio *Web* ao utilizador no decorrer da sessão.
- `Segundos de Duração da Sessão`: Tempo total, em segundos, de duração de uma sessão.
- `Segundos Para Servir Sessão`: Número total de segundos que tardou a servir todos os objectos numa sessão.
- `Segundos Visualização Sessão`: Tempo estimado, em segundos, que as páginas *Web* servidas nas sessão estiveram visíveis no navegador do utilizador.
- `Número de páginas Visitadas`: Indica quantas páginas foram servidas ao utilizador.
- `Número de Encomendas Submetidas`: Este valor diz-nos quantas encomendas foram submetidas pelo utilizador durante a duração da sessão.
- `Número de Produtos Encomendados`: Número de distintos produtos encomendados durante uma sessão.
- `Valor das Encomenda Submetidas em Euros`: Valor total das encomendas submetidas durante a sessão.

Poderão existir várias medidas do tipo monetário, com valores antes de imposto, valores do imposto, valores com descontos ou sem descontos, ou simplesmente o valor a pagar pelo cliente. Tudo dependerá das necessidades de análise existentes.

Capítulo 5

Extracção de Dados

O processo de *Extracção, Transformação e Integração* (ETI) no *Data Webhouse* conceptualmente não difere grandemente de um processo de ETI de um *Data Warehouse* mais "clássico". Todavia, a heterogeneidade das fontes de dados de *clickstream*, os problemas inerentes à tecnologia utilizada na *Web* e o eventual volume massivo de dados obrigam a especial atenção na construção e implementação do processo de ETI.

Posto de uma forma simples, o processo de extracção visa a recolha e transferência dos dados das suas diversas fontes para a *Zona de Concentração de Dados* (ZCD). A ZCD assume um papel essencial no processo de ETI. Para além de funcionar como ponto centralizador de dados, assume também outros papéis. É o sítio para onde os dados são transferidos, limpos, combinados, transformados e armazenados até estarem prontos a ser integrados no *Data Webhouse*. Funciona ao mesmo tempo como repositório tanto de meta-dados como de tabelas auxiliares, de repositório do estado dos diversos processos de ETI e de armazenamento de dados temporários ou intermédios.

O processo de ETI pode ser observado no exemplo apresentado (Figura 5.1). Neste exemplo é dado especial revelo para a componente de processamento dos dados de *clickstream*. Este processo de actualização de dados no *Data Webhouse* poderá ser definido como um *workflow* cujas actividades dependem da disponibilidade dos dados para extracção, limpeza e integração, e onde o despoletar dessas actividades será condicionado pelas necessidades de qualidade e

actualidade dos dados [Bouzeghoub et al. 99]. Os sistemas de gestão de *workflow* melhoram a eficiência dos processos através da coordenação automatizada dos dados e recursos necessários para a execução de actividades individuais [Muehlen01]. Para implementação do processo de ETI através um *workflow* podemos optar pelo uso dos *Data Transformation Services* (*DTS*) [MSSQLBooks04] caso o *Data Webhouse* seja implementação no SGBD *Microsoft SQL Server* 2000. Se a opção for por um SGBD Oracle então poderá ser utilizado o *Oracle Warehouse Builder* [OWB04]. Existem, contudo, várias decisões que têm de ser tomadas por forma a poder implementar este processo. Iremos ver quais ao longo dos próximos capítulos.

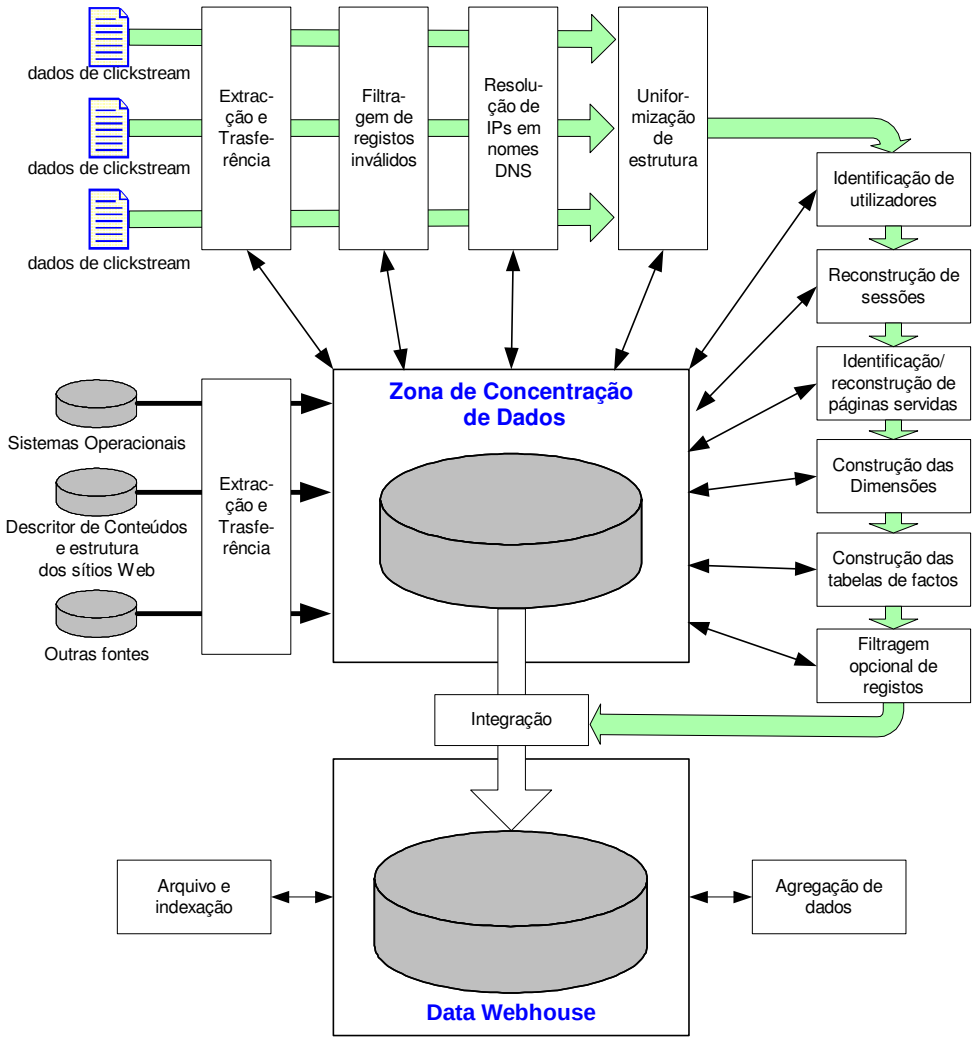

Figura 5.1 – Fluxo do processo de ETI para um *Webhouse*

5.1 Métodos e Mecanismos de Colecta

Quando estamos perante um cenário que inclui fontes de dados externas e internas à organização torna-se bastante complexo efectuar um controlo rigoroso do processo de extracção de dados. O nível do controlo a aplicar sobre as fontes varia em termos dos processos de extracção. Enquanto que a decisão de quando e como a extracção de dados pode ser feita é mais flexível, em fontes de dados internas essa flexibilidade pode ser perto de zero em fontes externas. É apenas após a análise das diversas fontes de dados que a decisão de como a colecta de dados é efectuada pode ser tomada.

Por forma a suportar o aumento do tráfego *Web*, uma organização poderá replicar os seus servidores *Web* e criar o que normalmente se apelida de *Web Farm* ou *Web Cluster* [Cardellini et al. 02]. Uma opção é também contratar o alojamento do servidor *Web* a empresas especializadas no ramo. Embora esta decisão possa ser justificável em termos de custo ou desempenho tenderá, contudo, a aumentar o grau de complexidade do processo de extracção de dados. Os ficheiros de *log* dos diversos sítios *Web* terão de ser extraídos dos servidores onde estão alojados e unificados para posterior processamento.

A Zona de Concentração de Dados não se situa, muito provavelmente, no mesmo servidor do sítio *Web*. Haverá então que extrair os dados das fontes e reuni-los nesta mesma zona para que possam ser convenientemente trabalhados. A passagem de dados das fontes para o *Data Webhouse* poderá ser feita de duas maneiras: incremental ou completa. Se passarmos os dados de forma incremental, apenas teremos que migrar os dados novos ou os actualizados desde o último povoamento do *Data Webhouse*. A passagem de dados incremental propicia um melhor desempenho, visto que o volume de dados a tratar é claramente menor. Poderá, no entanto, ser impossível usar este tipo de migração de dados se tivermos que lidar com sistemas que não podem, ou não têm a capacidade, de manter um histórico das mudanças ocorridas nos seus dados. Nesse caso, a passagem de dados completa é a única opção.

Os dados de *clickstream* relativos a uma visita poderão não estar apenas nos ficheiros de *log* dos servidores *Web*. Podem estar espalhados, por exemplo, em *logs* de servidores de publicidade *on-line*, em *logs* de servidores *proxy* de *cache*, nos servidores *Web* aplicacionais ou multimédia. Somente depois de reunir os dados de *clickstream* de todas estas fontes se poderá reconstruir fidedignamente a sessão do utilizador. Este processo é completamente dependente de quando os dados são transferidos das diversas fontes. Os dados transferidos dos diversos sistemas deverão reflectir o mesmo período de análise: uma hora, um dia, etc. Os sistemas que geram estes *logs* deverão ser configurados, se possível, para terem o mesmo período de rotação para os *logs*: um ficheiro de *log* por hora, um ficheiro de *log* por dia, etc. Quando não é possível implementar um sistema de rotação de ficheiros, por exemplo quando existe um único ficheiro de *log*, então deverá ser considerado o uso de sistemas de redireccionamento de dados – *piping*. Caso nenhum dos métodos descrito possa ser implementado então poder-se-á, em último recurso, copiar o ficheiro de *log* na sua totalidade e remover os dados não relevantes ou duplicados antes de iniciar o seu processamento. No caso dos ficheiros de *log* de servidores Microsoft IIS, com o formato Microsoft W3C Extended, poderá ser definido um período de rotação que determina o momento e a posição em que devem ser despoletados os processos de extracção de dados. Se esse período for diário, então serão gerados ficheiros que seguem a nomenclatura "`ex<YYMMDD>.log`" [MicrosoftIIS]. Por exemplo, para a data `2003-06-18` o ficheiro de *log* tomaria o nome de "`ex030618.log`". Quanto ao servidor *Web Apache*, este não tem, por omissão, nenhum sistema de rotação de ficheiros de *log* implementado. Todos os acessos HTTP ao servidor ficam tipicamente registados num único ficheiro de nome "`access_log`". Pode, no entanto, ser também configurado para efectuar uma rotação de *logs* recorrendo ao programa "`rotatelogs`", incluído na sua versão 2 [ApacheHTTP]. Nesta configuração podemos também definir o tempo de rotação e o nome que cada ficheiro vai tomar, por exemplo "`cex<YYMMDD>.log`". Para que os nomes reflictam realmente o conteúdo do *log* há que garantir também que a data no nome do ficheiro é relativamente a GMT+0. Se assim o não for, podemos incorrer em discrepância temporal de conteúdos. Enquanto com o Microsoft IIS os nomes dos ficheiros já reflectem a data GMT com o *Apache* esta é mais uma parametrização possível como o "`rotatelogs`".

A selecção do método de transferência de dados é também uma das decisões que tem de ser tomada. Para tal podemos usar dois métodos diferentes: *push* ou *pull* [Sweiger et al. 02] [Bouzeghoub et al. 99]. Com métodos *push* os processos de transferência de dados para a ZCD

são iniciados nos sistemas onde os dados são gerados. No caso da utilização de um método *pull* o processo desenrola-se de forma oposta aos métodos *push*. Existirá um processo na ZCD que será o responsável pelo início da colecta e transferência dos dados. A decisão sobre qual dos métodos a usar depende do tipo da fonte de dados e do pré-processamento necessário. Ao usar um método do tipo *pull* o controlo e monitorização do processo de extracção é, em princípio, simplificado já que é feito a partir de um ponto centralizado. Todavia, a capacidade de execução remota de comandos fica de certa forma mais limitada. No caso de extracção de dados de, por exemplo, uma *Firewall* por *hardware*, um método de *pull* teria de ser usado já que esta, provavelmente, não teria a capacidade por si própria de iniciar uma transferência. Se houver a necessidade, e capacidade, de executar operações, tais como comprimir ficheiros de *log* ou detectar mudanças no formato dos ficheiros de *log*, então poderá ser utilizado um agente de carregamento, residente no mesmo servidor da fonte de dados, e uma transferência do tipo *push* implementada. A existência deste agente de carregamento poderá também facilitar a detecção automática de mudança das fontes, tanto em conteúdo como em estrutura. Caso existam mudanças, este agente poderá despoletar um conjunto de acções pré-programadas que podem passar, entre outros, pela adaptação do método de transferência ou processamento ou mesmo pela notificação do administrador do sistema. Em ambos os métodos, *push* ou *pull*, há que considerar os seguintes aspectos:

- Quando é que os processos de extracção podem correr por forma a não causarem demasiado impacto na normal operação do sistema.
- Como monitorizá-los e verificar o seu sucesso.
- Como efectuar a notificação da ocorrência de situações de erro aos processos na ZCD.

São vários os mecanismos que temos para realizar as operações de extracção e transferência de dados. No caso dos ficheiros de *log* dos servidores *Web* os mecanismos de colecta poderão ser implementados sobre protocolos como o *File Transfer Protocol* (FTP), ou através de comandos normais de cópia de ficheiros suportados por sistemas de partilha de disco, como é o caso do *Network File System* (NFS) ou do Microsoft Windows *share*. Estes serão, todavia, mais indicados para sistemas internos à organização onde o nível de segurança necessário na transferência não é, tipicamente, muito elevado. No caso de ser necessário efectuar a colecta dos dados de forma segura, poderemos utilizar programas como o *Secure* FTP (SFTP), *Secure Copy* (SSC) ou utilizar transferências via HTTPS.

Se houver a necessidade de recolher dados em sistemas de bases de dados relacionais, como, por exemplo, uma base de dados de descrição da estrutura e conteúdo do sítio *Web*, teremos que apelar a outro tipo de mecanismos de extracção. Podemos ter, como alternativa, *triggers* de bases de dados, processos residentes que podem ser chamados por RPCs, ligações directas estabelecidas entre tabelas ou replicações para a zona de concentração de dados que iniciam a transferência de dados sempre que necessário.

Poderão sempre existir aplicações que trabalhem com sistemas de dados inacessíveis ou cuja estrutura de armazenamento seja desconhecida. Se uma funcionalidade de exportação de dados não estiver presente, então uma maneira de extrair dados será a de capturar o conteúdo dos vários métodos de saída de dados dessas aplicações, tais como listagens e relatórios *standard* .

5.2 Questões Resultantes do Conteúdo e Estrutura dos *logs*

A maioria dos servidores *Web*, servidores *Proxy* de *Cache* e *Firewalls* têm a tradução de endereços IPs em nomes fornecidos pelos servidores de DNS desactivados. Isto deve-se ao facto desta tradução acarretar um acréscimo do tempo de processamento. Desta forma, nos ficheiros de *logs* apenas estará presente o endereço IP do visitante. Caso se traduza este endereço para o respectivo nome, haverá um acréscimo da informação obtida. Conseguiremos obter, por exemplo, a informação sobre qual o país do fornecedor de acesso à Internet do visitante.

Com a utilização de serviços de DNS dinâmicos poderá ocorrer que um endereço IP seja traduzido para nomes distintos em diferentes ocasiões. Devido a este facto, é aconselhável que a tradução dos endereços IP seja feita o mais próximo possível do momento em que foi registado o pedido HTTP nos *logs* dos servidores *Web*. Desta forma poder-se-á evitar a incorrecta identificação do computador e domínio do utilizador visitante.

Quando existem vários sítios *Web* alojados no mesmo servidor então estes deverão ser identificados claramente em cada entrada no ficheiro de *log*. Isto porque o comportamento de

alguns servidores *Web* poderá ser o de registar os acessos a todos os sítios *Web* num único ficheiro de *log*.

A identificação do sítio *Web* deverá também ser feita nos ficheiros quando um Webhouse é usado para analisar dados de múltiplos sítios *Web* mesmo quando estes se encontram em servidores diferentes. Esta identificação irá facilitar a correcta identificação, filtragem e processamento dos dados. Enquanto que o formato *standard* de *log* da W3C considera a existência de um atributo específico para esta identificação, o mesmo não é considerado pelos formatos *NCSA Common* e *Extended Log Format*. Caso seja possível, os servidores *Web* que usem os formatos da NCSA deverão aumentar o número de atributos registados no *log* por forma a que esta identificação do sítio *Web* possa ser feita. Em último recurso, o registo dos pedidos HTTP deve ser feito individualmente: um ficheiro de *log* por cada sítio *Web*.

Os dados de *clickstream* provenientes de fontes heterogéneas são propensos à ocorrência de problemas ao nível do esquema e de instância. Cada fonte poderá registar os dados de *clickstream* em ficheiros de *log* com formatos diferentes e com os mesmos conteúdos representados de forma distinta. Vejamos então alguns dos possíveis problemas ao nível de instância e estrutura que podemos encontrar. Suponhamos que uma organização tem dois servidores:

- O primeiro, servidor A, corre um servidor *Web Apache* e regista os pedidos HTTP no *NCSA Extended Log Format* (`remotehost, ident, authuser, request, status, referrer, user-agent`).
- O segundo, servidor B, corre o Microsoft IIS e regista os pedidos HTTP num ficheiro com o formato *Microsoft W3C Extended Log Format* com os seguintes campos: `date, time, c-ip, cs-username, s-ip, cs-method, cs-uri-stem, cs-uri-query, sc-status, sc-bytes, time-taken, cs-version, cs(User-Agent), cs(Cookie)` e `cs(Referer)`.

Tomando como exemplo um utilizador que acedeu a uma página no servidor A e nessa página seleccionou um apontador que o levou para uma página no servidor B. A entrada no ficheiro de *log* do servidor A é a seguinte:

```
10.32.100.2 - - [04/May/2003:19:08:48 +0100] "GET /inicio.htm
HTTP/1.1" 200 818 "-" "Mozilla/4.0 (compatible; MSIE 5.0; Windows
95; DigExt)"
```

E a entrada registada no *log* do servidor B é:

```
2003-05-04 18:08:59 10.32.100.2 - 10.32.100.1 GET
/Liverpool/liverpool.htm - 200 707 110 HTTP/1.1
Mozilla/4.0+(compatible;+MSIE+5.0;+Windows+95;+DigExt) -
http://www.servera.org/inicio.htm
```

Observando ao pormenor as entradas registadas nos *logs* dos dois servidores podemos detectar a existência de vários conflitos a nível de nomenclatura e estrutura. Se, por qualquer razão, estes conflitos não conseguirem ser corrigidos a nível das fontes então será necessário efectuar mapeamento, transformação e integração dos esquemas por forma a podermos criar um formato neutral que possa ser trabalhável [RahmDo00].

O campo `date` registado no *log* do servidor A e o campo `date` registado no *log* do servidor B são campos parcialmente homónimos. No primeiro caso, o campo `date` é representado entre parêntesis rectos, '[' e ']'. Este campo para além do valor da data também inclui a hora do pedido HTTP e a compensação temporal relativa à hora GMT. No segundo caso, o campo `date` apenas contém um valor que realmente reflecte uma data de calendário. O esquema final teria de evoluir a partir destas duas representações para um esquema onde houvesse uma representação inequívoca dos conteúdos dos campos. Poderíamos ter no esquema final, por exemplo, dois campos para a informação temporal do pedido http: "`Data GMT`" e "`Hora GMT`".

Existe um problema a nível de instância também relacionado com a representação do valor das datas. No caso das datas registadas no *log* do servidor A, a data segue o formato `DD/MÊS/AAAA` (ex: `04/May/2003`) dois dígitos para representar o dia, três letras para representar o mês e quatro dígitos para representar o ano. Contudo no *log* do servidor B o formato seguido para a data é o de `AAAA-MM-DD` (ex: `2003-05-04`) quatro dígitos para representar o ano, dois dígitos para representar o mês e dois dígitos para representar o dia. Esta representação teria também de

evoluir para o mesmo formato podendo, por exemplo, evoluir para uma representação igual ao segundo exemplo já que o primeiro exemplo tem uma representação textual do mês baseado na língua Inglesa. Existem também casos de sinónimos nos dois formatos de *log* usados pelos servidores. O campo `remotehost` na entrada registada no *log* do servidor A é um sinónimo do campo `c-ip` registado no *log* do servidor B. Voltando a analisar o exemplo dos dois registos nos ficheiros de *log* podemos notar mais uma inconsistência na representação da informação que identifica qual foi o agente HTTP, tipicamente o navegador, usado pelo utilizador. No caso do *NCSA Extended Common Log Format*, usado pelo servidor A, esse agente é representado por:

```
"Mozilla/4.0 (compatible; MSIE 5.0; Windows 95; DigExt)"
```

O formato usado pelo servidor B é *o Microsoft W3C Extended Log Format* e, neste caso, a representação do mesmo agente do utilizador é feita da seguinte forma:

```
Mozilla/4.0+(compatible;+MSIE+5.0;+Windows+95;+DigExt)
```

As diferenças são simples de detectar. Enquanto no primeiro caso o texto correspondente ao agente do utilizador está delimitado por aspas (") no segundo caso isso não acontece. Como a separação entre campos no *Microsoft W3C Extended Log Format* é feita pelo caracter de espaço então todos os espaços que possam existir dentro do texto que representa o agente são substituídos pelo sinal de mais (+). O tratamento deste texto tem de passar por uma substituição de todos estes sinais de mais pelo respectivo caracter de espaço. De forma semelhante os caracteres no URI que foram codificados pelo servidor *Web* terão de ser substituídos pelo caracter correcto. Por exemplo, se no texto passado como parâmetro no URI constasse um texto igual a:

```
category=SD%20Memory%20Cards
```

este mesmo texto teria de ser sujeito a uma operação de limpeza e passar a ser representado pelo respectivo caracter ASCII, ou seja, pelo caracter que representa o espaço:

```
category=SD Memory Cards.
```

Várias filtragens sobre os dados constantes nos *logs* poderão também ser feitas. Deverão ser aplicados filtros com expressões regulares com o objectivo de extrair os registos que se apresentam no formato esperado. Isto poderá ser feito simplesmente com um guião desenvolvido em `awk` ou em `Perl`. A verificação da validade da sintaxe do URI segundo a especificação apresentada no RFC2616 [Fielding et al. 99] pode ser também forçada. Contudo, há que acautelar que o servidor *Web* poderá usar notações não *standard* no URI para, por exemplo, efectuar a passagem de parâmetros. Para a correcta interpretação destes casos será necessário um conhecimento semântico dos URIs [Baglioni et al. 03].

Se nos objectivos do *Data Webhouse* não constar a análise de, por exemplo, ataques conhecidos ao sítio *Web* então outra filtragem poderá incluir todos aqueles registos que representam esse tipo de tráfego. O mesmo poderá ocorrer com as visitas efectuadas por agentes automáticos e deixar apenas os acessos efectuados pelos utilizadores humanos. A estratégia que normalmente se adopta na identificação deste tipo de acessos passa pela comparação de dados registados no *log* contra listas de características conhecidas desses pedidos HTTP. Mais à frente aprofundar-se-á a discussão da identificação destes agentes automáticos.

5.3 Uniformização de Estrutura

No final do processo de extracção, todos os dados de *clickstream* provenientes dos distintos ficheiros de *log* deverão ser apresentados num esquema com uma estrutura uniformizada. Esta deverá conter toda a informação possível de extrair dos ficheiros de *log* em campos distintos. Nesta estrutura uniformizada todos os problemas com sinónimos, homónimos e de conflitos de esquema devem ficar resolvidos.

Um exemplo de uma possível estrutura pode ser vista na tabela apresentada (Tabela 5.1). No caso das *cookies* é sugerido a existência, em função da utilização dada pelos programadores dos sítios *Web*, de vários campos que incluam o nome e valor das *cookies* existentes.

Campo	*Descrição*
Data	Data em que o pedido terminou de ser servido (GMT). Formato AAAA-DD-MM.
Hora	Hora em que o pedido terminou de ser servido (GMT). Formato HH24:MM:SS.
Endereço IP	Endereço IP do cliente.
Nome do computador do visitante	O nome, retornado pelos servidores de DNS, da máquina do visitante.
Nome do utilizador	Nome do utilizador, caso a autenticação seja feita pelo servidor *Web.*
Serviço e Instância	Nome do sítio/serviço no servidor.
Nome do servidor	Nome do servidor onde o ficheiro de *log* foi gerado.
Endereço IP do servidor	Endereço IP do servidor onde o ficheiro de *log* foi gerado.
Porta	Porta IP onde o navegador do cliente ligou (normalmente 80 para HTTP ou 443 para https).
Método	Acção executada, por exemplo um GET, um HEAD ou um POST.
URI Stem	O caminho dentro do servidor *Web* para o objecto acedido.
Parâmetros do URI	A componente de parâmetros incluída no URI após o caracter "?".
Estado http	O código de estado HTTP resultante, segundo RFC2616 [Fielding et al. 99].
Bytes enviados	Número de *bytes* enviados pelo servidor *Web.*
Bytes recebidos	Número de *bytes* recebidos pelo servidor *Web.*
Tempo demorado	O tempo que a acção demorou a ser executada, em segundos.
Nome do protocolo	O nome do protocolo usado pelo navegador do cliente. Por exemplo HTTP.
Versão do protocolo	Por exemplo 1.0 ou 1.1.
Agente* HTTP *do visitante	A aplicação utilizada no computador do visitante que emitiu o pedido. Normalmente identifica o navegador *Web.*
Sítio Referenciador	Indica por quem o pedido do visitante diz ter sido referenciado.
URI Stem do Referenciador	Este valor indica qual foi a página por quem o pedido do visitante diz ter sido referenciado.
Parâmetros do URI do Referenciador	A componente de parâmetros incluída no URI do referenciador após o caracter "?".
Nome Cookie 1	O nome da *cookie 1* enviada pelo cliente HTTP, se esta existir.
Valor Cookie 1	O valor da *cookie 1* enviada pelo cliente HTTP, se esta existir.
Nome Cookie 2	O nome da *cookie 2* enviada pelo cliente HTTP, se esta existir.
Valor Cookie 2	O valor da *cookie 2* enviada pelo cliente HTTP, se esta existir.

Tabela 5.1 – Estrutura Uniformizada de ficheiros de *log*

Os dados contidos nesta estrutura são directamente dependentes dos conteúdos dos ficheiros de *log*. Esta estrutura poderá ser expandida para, por exemplo, incluir dados provenientes de servidores multimédia. Aqui, no entanto, foram apenas considerados dados que se poderão obter a partir dos formatos de *logs* de servidores *Web* mais comuns.

Haverá um momento em que será necessário efectuar a passagem dos dados que existem a nível de ficheiros para um SGBD. Nesse momento será recomendável a utilização de um programa de carregamento em bloco tal como o `sqlload` em SGBD Oracle ou o `bcp` em SGBD *Microsoft SQL Server*, ambos invocáveis a partir da linha de comando do sistema operativo. O ficheiro com a estrutura uniformizada poderia também, por exemplo, ser carregado para uma tabela em *Microsoft SQL Server* de nome `tabela_weblogs`, com uma estrutura igual à do ficheiro, com uma instrução em Transact-SQL semelhante ao seguinte exemplo:

```
BULK INSERT [Tabela_Weblogs] FROM 'c:\iislogs\novo\20040501.log'
WITH (
    FIELDTERMINATOR = ' ',
    ROWTERMINATOR = '\n'
)
```

Neste caso, o nome do ficheiro representaria a data do seu conteúdo seguindo um formato `aaaammdd.log` com quatro dígitos para representar o ano, dois dígitos para o mês e outros dois para o dia. Convém ter em atenção que, no caso de ficheiros gerados no formato W3C, esta instrução falharia devido à existência das linhas com directivas, iniciadas pelo caracter '#'. Estas linhas, no caso do servidor *Web* Microsoft IIS, são adicionadas sempre que existe uma rotação do ficheiro de *log* ou quando o servidor é reiniciado. Estes caracteres deverão ser retirados do ficheiro antes da operação de carregamento através de um pequeno utilitário disponibilizado pela Microsoft de nome `PrepWebLog` [MicrosoftKB296093]. Ou com um simples filtro implementado com comandos existentes a nível do sistema operativo como o `sed` ou com guiões escritos em `Awk` ou `Perl`. A Microsoft também disponibiliza um interpretador de *logs* capaz de inserir ficheiros de *log*,

em diversos formatos, directamente no *Microsoft SQL Server* para além de permitir executar interrogações do tipo SQL directamente sobre ficheiros de *log* [MicrosoftIIS6RKT03].

Capítulo 6

Transformação de Dados de *Clickstream*

Em termos gerais, durante a fase de transformação de dados são realizadas operações de junção e associação sobre os dados alvo, deixando de fora eventuais registos que não são necessários no *Data Webhouse*. Tenta-se, assim, reduzir tanto quanto possível, o volume de dados envolvidos sem, no entanto, pôr em causa a sua integridade e significado necessários à obtenção do grão definido para as tabelas de factos.

6.1 Identificação de Utilizadores

A identificação de utilizadores deverá ser feita por forma a que lhes seja associado um identificador único que os permita identificar em visitas posteriores. Se não for possível, então deverão ter um identificador único de visitante anónimo. Existe, contudo, uma dificuldade na distinção entre utilizadores resultante da definição do protocolo HTTP [BernersLee et al. 96] [Fielding et al. 99] e da informação registada nos *logs* sobre o autor de cada pedido HTTP.

São várias as possíveis técnicas para distinguir entre utilizadores (Tabela 6.1). Todavia na escolha de qual a técnica a usar, quando a escolha é possível, deverão ser pesadas as vantagens e desvantagens associadas.

A técnica mais simples passa pela obrigatoriedade do utilizador se autenticar para ter acesso aos conteúdos do sítio *Web*. Se essa autenticação for feita pelo servidor *Web* então o nome com que o utilizador se autentica ficará registado nos *logs* do servidor *Web* – campo `cs-username` no formato *Microsoft W3C Extended Log Format* ou campo `authuser` no *NCSA Common Log Format* e *NCSA Extended Common Log Format*. Se a autenticação for a nível aplicacional então terá de haver um mecanismo que relacione a informação registada no *log* com o controlo de acessos feito na aplicação *Web*. Isto para poder correctamente identificar quem se autenticou na altura em que estiver a ser feito o processamento dos *logs*, por forma a distingir entre os diversos utilizadores. Este mecanismo de ligação poderá ser um identificador registado numa *cookie* ou adicionado ao URI.

Se a autenticação não for exigida logo no primeiro pedido do utilizador então não se saberá quem pediu as páginas previamente servidas. Isto por não terem a identificação do utilizador a quem foram servidas registada no ficheiro de *log*. Muitos utilizadores são, todavia, relutantes em efectuar qualquer tipo de registo e evitam assim os acessos a sítios que tal o exijam. Ora isto terá de ser pesado na altura de decidir sobre a utilização deste método de rastreio de sessão já que *per si* poderá ser um factor dissuasor da utilização do sítio *Web* e, consequentemente, prejudicial ao objectivo do mesmo.

Se a autenticação não for necessária então o identificador de utilizador poderá ser atribuído automaticamente pelo servidor *Web*, ou programaticamente pelas aplicações *Web*. Neste caso, será apenas uma distinção entre utilizadores anónimos.

O uso de *cookies* persistentes para o armazenamento do identificador de utilizador terá a vantagem de permitir identificar visitas posteriores do mesmo utilizador. Este método é, todavia, falível como elemento de distinção entre utilizadores, caso o mesmo computador e navegador sejam utilizados por mais de um utilizador.

Caso o uso de *cookies* seja rejeitado pelo navegador do utilizador então o servidor, ou aplicações *Web*, deverão passar este identificador do utilizador para o URI. Se assim for, cada vez que o utilizador visitar o sítio *Web*, este será identificado como um novo utilizador.

Método	***Vantagens***	***Desvantagens***
Obrigatoriedade do utilizador se autenticar para obter acesso ao sítio	Conhece-se a pessoa e não apenas o computador que acedeu ao sítio *Web*.	Dissuasor de utilização do sítio bem visto que muitos utilizadores não estão dispostos a se registarem.
Identificador de utilizador atribuído automaticamente pelo servidor Web	Se armazenado em *cookies* permanentes permite identificar visitas posteriores.	Se vários utilizadores usarem o mesmo navegador então este método não os consegue distinguir.
Variáveis nos cabeçalhos de pedidos http: ***Accept*** ***Accept-Charset*** ***Accept-Encoding*** ***Accept-Language*** ***User-Agent***	Parte do *standard* HTTP e como tal estarão sempre disponíveis.	Apenas utilizáveis se o servidor *Web* conseguir registar os seus valores no ficheiro de *log*. Não estão, na sua totalidade, disponíveis nos formatos *standard* de *logs* de servidores *Web*. Se vários utilizadores usa-rem o mesmo navegador então este método não os consegue distinguir.
IP + agente http	Sempre disponíveis e registados na maior parte dos ficheiros de *log standard* .	A combinação não é válida no caso de endereços IPs rotativos ou de substituição ou então se são usados múltiplos agentes. Se vários utilizadores usarem o mesmo navegador então este método não os consegue distinguir.
Valores de configurações do navegador e ou computador do visitante passados no URI e que ficam registados no log do servidor Web (ex.: hora local, resolução gráfica, profundi-dade da cor, etc.)	Permitirem melhorar o processo de distinção entre utilizadores, ou navegadores, diferentes. Fornecem elementos adicionais passíveis de serem utilizados para melhorias técnicas do sítio *Web*.	Agentes HTTP poderão não suportar, ou não querer, executar os programas que obtenham e transmitam essa informação no URI. Deverá funcionar conjunta-mente com outro método de distinção entre utilizadores.

Tabela 6.1 – Elementos de distinção entre utilizadores

As variáveis passadas nos cabeçalhos dos pedidos HTTP [Fielding et al. 99] permitem ao agente HTTP transmitir ao servidor informação adicional acerca do pedido, bem como acerca dele próprio. Dentro das diversas variáveis transmitidas nos cabeçalhos, serão aquelas que fornecem informação específica sobre a configuração do navegador que, eventualmente, maior utilidade terão na distinção entre utilizadores.

A utilização da combinação entre o endereço IP e o agente HTTP poderá não resultar, visto que o mesmo endereço IP poderá ser usado por múltiplos utilizadores. A causa do endereço IP ser usado por vários utilizadores pode ser a utilização de servidores *proxy* que actuam como servidores de autenticação, *cache* de conteúdos, ou simplesmente como supressores de identidade. Nesses casos, tipicamente, será o endereço IP do servidor de *proxy*, servindo múltiplos pedidos e utilizadores, que chega ao servidor *Web*. Um outro problema põe-se quando temos a utilização de múltiplos endereços IP numa sessão por um único visitante. Ocorre, por exemplo, quando um fornecedor de acesso à Internet dispõe de vários servidores *Proxy* com tarefas específicas, por exemplo, umas para *cache* de imagens, outras para páginas html, etc. Temos também as situações onde existem *firewalls* que substituem todos os endereços IP dos computadores na rede por ela protegida, eventualmente endereços IP não válidos em redes públicas, por um outro endereço IP, porventura o seu próprio. Uma heurística passível de ser usada para distinguir entre utilizadores que têm o mesmo par "IP + agente HTTP" é sugerida em [Cooley et al. 99]. Nesse estudo, é sugerido que se uma página pedida não for directamente acessível a partir de uma página já anteriormente servida ao utilizador então poder-se-á assumir que se está na presença de um outro utilizador com o mesmo endereço IP. Para ser possível este tipo de suposição é necessário recorrer à análise do referenciador de cada pedido e ter presente a descrição da estrutura e conteúdo do sítio *Web*.

O último elemento de distinção entre utilizadores descrito (Tabela 6.1) funcionará mais como um complemento ao uso, por exemplo do "IP + agente HTTP", e não como elemento individual de distinção. Em casos onde existam vários utilizadores da mesma rede, porventura com o mesmo agente HTTP e com um endereço IP de uma *firewall*, a efectuarem pedidos ao sítio *Web* as configurações locais efectuadas por cada utilizador, tais como hora local no seu computador, resolução gráfica, profundidade da cor, etc. serão um elemento adicional de distinção.

A alternância da utilização entre agentes HTTP também é um factor que poderá deturpar a validade da combinação "IP + agente HTTP". Poderão existir situações em que, após verificar que o seu navegador preferido não consegue exibir correctamente as páginas de um sítio *Web*, o utilizador alterna para um outro navegador que tenha instalado no seu computador e para o qual o sítio *Web* esteja optimizado.

Em máquinas de acesso público, como por exemplo em *Cyber* Cafés, temos também o problema de que o utilizador de um computador, eventualmente com o par "IP + agente HTTP" fixo, muito provavelmente, não será o mesmo utilizador num momento diferente, já que um mesmo computador é partilhado por múltiplos utilizadores.

6.1.1 Agentes Automáticos

Nem todas as visitas ao sítio *Web* são, todavia, feitas por utilizadores de "carne e osso". Serão muitos os casos em que os pedidos efectuados ao sítio *Web* são feitos por agentes automáticos. Em [TanKumar02] é apresentado um caso de estudo onde em média cerca de 5% das visitas eram originadas por agentes automáticos. Contudo, esse valor representava cerca de 85% das páginas html servidas. Os valores apresentados em [KohaviParekh03] vão mais longe e indicam que os agentes automáticos podem gerar entre 5% a 40% das vistas registadas num sítio *Web*. Estes agentes automáticos são conhecidos por *robots*, *Webbots*, *spiders,* w*orms*, *Web crawlers, Web ants, wanderers* e *harvesters* entre outros. Os objectivos destes programas são vários:

- Indexação.
- Validação de páginas HTML.
- Validação de apontadores.
- Alertar utilizadores sobre nova informação disponível no servidor.
- Replicação de conteúdos entre servidores.
- Descarregar conteúdos do servidor *Web* para o computador do utilizador para permitir uma consulta quando desconectado da Internet.
- Recolha de emails para serem usados como alvos de campanhas de email não solicitado.
- Contactos com o servidor *Web* para recolha de dados para elaboração de estatísticas diversas.

Generalizando, poderemos considerar por **agente automático** qualquer programa que percorre e obtém automaticamente documentos na *Web* através da utilização do protocolo HTTP, quer seja seguindo os apontadores que encontra quer por qualquer outro método [ChauChen03].

Enquanto as visitas dos robots de indexação são desejáveis devido aos utilizadores que os seus respectivos donos, entenda-se os motores de pesquisa, poderão referenciar, muito do tráfego que os restantes agentes automáticos geram poderá não contribuir para o objectivo do sítio *Web*. Este tipo de programas pode provocar um grande número de registos nos *logs* do servidor *Web* que poderão falsear as análises efectuadas. A correcta identificação dos pedidos como tendo sido feitos por agentes automáticos será importante. Será em função dessa identificação que correctamente se poderá seleccionar ou filtrar os seus registos nos ficheiros de *log* do servidor *Web*. Esta selecção ou filtragem deverá ser feita em função do objectivo do *Data Webhouse* e do interesse dos seus utilizadores finais.

O tráfego por parte dos agentes automáticos poderá, à partida, tentar ser minimizado pelo sítio *Web* através de dois métodos diferentes. O primeiro toma o nome de "Protocolo de Exclusão de Robots" [Koster94] e define a existência do ficheiro `robots.txt` na raiz do servidor *Web* onde são dadas directivas aos agentes automáticos sobre o que pode ou não ser visitado dentro do sítio *Web*. No extremo, podemos negar o acesso completo, para fins de indexação, a todo o sítio *Web*. Para tal o ficheiro `robots.txt` deverá ter o seguinte conteúdo:

```
User-agent: *
Disallow: /
```

Tipicamente serão apenas indicados alguns dos directórios que não devem ser visitados. Com este método não se evita que haja, pelo menos, sempre um acesso por parte dos agentes automáticos necessário para efectuar a consulta do próprio ficheiro `robots.txt`. Este método, a ser usado, poderá também afastar possíveis visitantes do servidor *Web* ao impedir que certo, ou todo, o conteúdo do sítio *Web* não seja indexado pelos motores de pesquisa.

O segundo método de redução de tráfego gerado pelos agentes automáticos consiste na utilização de meta-etiquetas inseridas nos cabeçalhos das páginas html. Estas etiquetas permitem aos autores das páginas indicar se a página poderá ser indexada ou então usada para extrair apontadores. No exemplo seguinte vemos a etiqueta a utilizar para indicar que a página onde ela está inserida não deve ser indexada e nenhum apontador seguido:

```
<meta name="robots" content="noindex,nofollow">
```

Este método permite uma indexação selectiva de páginas mas para que a etiqueta seja lida é necessário, pelo menos, o acesso à página onde ela está inserida.

Todavia, nem todos os agentes automáticos seguem as directivas especificadas pelos dois métodos indicados. Mesmo que o tráfego gerado seja diminuído, ele continua a existir e é necessário identificá-lo correctamente por forma a poder ser feita, como já dito, uma correcta selecção e filtragem de dados.

Existem várias técnicas (Tabela 6.2) que podem ser utilizadas para identificar estes agentes automáticos. Se for feita uma análise do campo `User-Agent` registado nos *logs* dos servidores *Web*, podemos comparar esse valor contra uma lista de agentes automáticos conhecidos. Esta técnica não é possível de utilizar caso este valor não exista, quer seja porque o servidor *Web* não está configurado para o colocar no *log* ou porque simplesmente este valor não foi transmitido pelo próprio agente. Sendo assim, poder-se-á então comparar a origem do pedido, endereço IP ou domínio, contra uma lista de valores conhecidos para tentar obter uma identificação.

As comparações contra listas, contudo, para além do esforço de manutenção resultam apenas para casos onde estamos perante pedidos de agentes automáticos já conhecidos. Por outro lado, existem agentes automáticos que tentam dissimular a sua verdadeira identidade passando valores na variável `User-Agent` do protocolo HTTP iguais aos de um vulgar navegador *Web*. Existem também situações onde o valor transmitido na variável `User-Agent` do pedido HTTP não é constante ao longo de uma visita de um agente automático [TanKumar02]. Em caso de não identificação através destas duas primeiras técnicas os acessos ao ficheiro `robots.txt` poderão ser uma outra alternativa. A não ser que seja um utilizador a tentar obter mais alguma informação

para fins pouco claros, os acessos a este ficheiro são feitos exclusivamente por agentes automáticos. Poderá, então, ser considerado que todos os pedidos efectuados a partir de uma origem a quem o servidor *Web* anteriormente serviu o ficheiro `robots.txt` terão sido gerados por um agente automático.

Método	***Vantagens***	***Desvantagens***
Comparar o valor do campo User-Agent contra listas de valores conhecidos	Variável sempre disponível no cabeçalho do protocolo HTTP.	Nem todos os robots se identificam. Exige a manutenção de uma lista e apenas identifica agentes automáticos conhecidos
Comparar a origem dos pedidos contra listas de valores conhecidos	Permite identificações mesmo se valor do `User-Agent` for omisso.	Exige manutenção de lista. Apenas identifica agentes automáticos conhecidos.
Verificar pedidos ao ficheiro robots.txt	Permite detectar agentes automáticos previamente desconhecidos.	Nem todos os agentes que consultam o ficheiro **robots.txt** são agentes automáticos. Nem todos os agentes automáticos consultam o **robots.txt**.
Verificar pedidos a páginas "ratoeira" indicadas por apontadores invisíveis a utilizadores humanos	Permite detectar agentes automáticos previamente desconhecidos.	Implica a adaptação de uma, ou mais páginas, do sítio *Web*.
Identificar agentes que nunca se autenticam	Permite detectar agentes automáticos previamente desconhecidos.	Aplicável apenas em sítios *Web* onde a autenticação é necessária. Falível com navegadores pouco comuns.
Identificando origem de pedidos com um grande número de pedidos* HTTP *que usam o método HEAD ou então que não tenham um referenciador	Permite detectar agentes automáticos previamente desconhecidos.	Pode ser falível porque também vulgares navegadores podem gerar pedidos com o método HEAD e sem referenciador.
Inferir a visita por um agente automático se existirem	Permite detectar agentes automáticos previamente	Pedidos genuínos podem ser incorrectamente identificados

acessos que não peçam todo o conteúdo das páginas (ex: não pede imagens)	desconhecidos.	como tendo sido feitos por agentes automáticos, é o caso de um navegador capaz de ler texto apenas ou com as imagens inactivas
Verificar pedidos constantes com intervalos curtos entre eles efectuados a partir da mesma origem	Permite detectar agentes automáticos previamente desconhecidos.	Robots que apenas verificam se ficheiro **robots.txt** mudou, ou um qualquer outro ficheiro, podem não ser detectados.

Tabela 6.2 – Métodos de identificação de agentes automáticos

O uso de páginas "ratoeira" desprovidas de conteúdo poderá ser também um método a utilizar. O acesso a estas páginas será feito a partir de apontadores invisíveis a utilizadores humanos inseridos em páginas *Web* normais. Se estas páginas forem consistentemente pedidas pelo mesmo tipo de agente então teremos um indicador que o agente que o fez será certamente um agente automático [KohaviParekh03]. A utilização deste método implica, contudo, a alteração de uma, ou mais, páginas do sítio *Web* para que incluam o apontador para a página "ratoeira". Em [KohaviParekh03] é sugerido que a maioria dos autores de agentes automáticos não inclui o suporte de conteúdo do tipo `gzip`. Este suporte teria, no entanto, de ser testado a nível aplicacional e em tempo real, pela análise da variável `Accept-Encoding` passada no cabeçalho HTTP, já que não é, normalmente, informação que fique registada nos ficheiros de *log* dos servidores *Web*.

Existem várias outras técnicas que entram pela análise dos padrões de navegação dos agentes. Em [TanKumar02] este caminho é explorado e é sugerido um esquema que recorre a modelos de classificação para detectar a presença de um agente automático.

6.2 Reconstrução de Sessões

Após a distinção entre os utilizadores ter sido efectuada, é, então, necessário definir quais os limites da sessão e identificar quando esta se inicia e termina.

De acordo com [W3C99] uma sessão, ou visita, é composta pelo conjunto de actividades efectuadas por um utilizador desde o momento em que este entra num sítio *Web* até ao momento em que ele o deixa. A reconstrução de uma sessão visa agrupar, sob um identificador único, todos os pedidos HTTP efectuados por um utilizador ao sítio *Web*. Dessa forma, essa visita poderá ser estudada e comparada com outras visitas, nomeadamente no que diz respeito ao seu resultado e se esta contribuiu ou não para o objectivo do sítio *Web*.

A reconstrução de uma sessão está intimamente ligada à identificação de distintos utilizadores, sendo que algumas das técnicas utilizadas contribuem para ambos os objectivos. Em situações onde se pretende efectuar o rastreio de utilizadores sem determinar a sua identidade, então o processo e técnicas utilizadas são praticamente idênticas às usadas para a reconstrução das sessões.

Existem essencialmente dois tipos de estratégias para efectuar a reconstrução de uma sessão: pró-activas e reactivas [Spiliopoulou et al. 03]. As estratégia pró-activas passam pela atribuição de um identificador de sessão único a cada utilizador antes ou durante a iteração do mesmo com o sítio *Web*. As estratégias reactivas tentam atribuir um identificador de sessão a cada pedido de um utilizador após estes terem ocorrido. Estes pedidos HTTP estarão espalhados ao longo do ficheiro, ou ficheiros, de *log* e incluem não só pedidos efectuados directamente pelo utilizador, como também os pedidos efectuados automaticamente a objectos incluídos nas páginas servidas. Não serão raros os casos onde o sítio *Web* faça uso de estratégias pró-activas mas, para a completa reconstrução de sessões, será também necessário aplicar estratégias reactivas. Isto porque a atribuição de um identificador de sessão pró-activamente poderá falhar para alguns dos pedidos HTTP. No seu conjunto, o processo de reconstrução de sessões passará pela aplicação das várias técnicas, primeiro as pró-activas e posteriormente as reactivas, em vários passos sucessivos que irão atribuindo o identificador de sessão aos vários pedidos HTTP. Na melhor das situações, todos os registos dos ficheiros de *log* acabarão com um identificador atribuído que os inclui na sessão correcta. Contudo, o processo poderá eventualmente falhar na inclusão de alguns registos dentro da sessão correcta devido à omissão de informação ou existência de informação incompleta nos dados de *clickstream*.

6.2.1 Estratégias Pró-activas

O caso mais simples de uma reconstrução de sessão ocorre quando o servidor *Web* atribui automaticamente um identificador de sessão a todos os pedidos HTTP, ou quando esta mesma atribuição é feita programaticamente pelas páginas do sítio *Web*. Neste caso, não é necessária a atribuição posterior de um identificador de sessão ao pedido, já que este fica automaticamente registado nos ficheiros de *log* quando o pedido HTTP é servido. A reconstrução de sessão passa apenas pela junção dos registos com o mesmo identificador. Há que considerar, no entanto, que a atribuição deste identificador de sessão pelo servidor *Web* poderá não ser único. Este é o caso, por exemplo, se essa atribuição for efectuada com recurso ao objecto de gestão de sessões disponibilizado pelo servidor Microsoft IIS v5 para aplicações desenvolvidas com *Active Server Pages* (ASP) [MicrosoftIIS5RG00]. Nesta situação, outro método de atribuição do identificador de sessão terá de ser utilizado por forma a garantir a unicidade do mesmo. Caso contrário, correr-se-ia o risco de ter erradamente associados a uma sessão pedidos HTTP efectuados por utilizadores distintos.

Em situações onde o sítio *Web* corre sobre vários servidores *Web* replicados, em *Web Farms* ou *Web Clusters*, a atribuição do identificador de sessão não poderá ser feita automaticamente pelo servidor *Web*. Esta atribuição terá de ser controlada ao nível aplicacional, com recurso ao armazenamento dos valores dos identificadores de sessão em repositório centralizado [Papa00].

Os identificadores de sessão são, tipicamente, armazenados em *cookies*. Devido ao funcionamento em modo de pedido e resposta do protocolo HTTP, será apenas após a primeira resposta do servidor *Web* que uma *cookie* poderá ser passada ao navegador do utilizador. Como tal, o registo no ficheiro de *log* do primeiro pedido efectuado pelo utilizador nunca terá informação da *cookie* que contém o identificador de sessão. Essa *cookie* apenas passará a ser registada em pedidos posteriores e, como tal, a identificação da primeira página visitada na sessão ficará comprometida. Uma estratégia comum para colmatar esta limitação do primeiro pedido será a do uso de uma página *Web* desprovida de conteúdo simplesmente com o objectivo de atribuir a *cookie* ao navegador do visitante. Esta página recebe os pedidos do utilizador e, caso este ainda não tenha um identificador de sessão, então responde ao pedido com a atribuição de uma *cookie* com o valor do identificador. Simultaneamente redirecciona o navegador para a página seguinte.

O uso de *cookies* poderá não estar, todavia, autorizado no navegador do visitante ou o utilizador poderá a meio de uma visita negar o uso das mesmas. Nesta situação, o sítio *Web* terá de estar preparado para recorrer à colocação do identificador de sessão embebido nos parâmetros do URI. Em [Sweiger et al. 02] é descrita uma forma de detectar se o navegador do utilizador tem ou não o uso de *cookies* autorizado. Se ambos os mecanismos de identificação de sessão, seja pelo uso de *cookies* seja por variável embebida no URI, forem usados no sítio *Web* então o processo de reconstrução de sessão terá que, obviamente, ser programado para analisar as duas possíveis localizações do identificador de sessão quando estiver a processar os ficheiros de *log*.

O uso de agentes embebidos nas páginas *Web* servidas, ou eventualmente um navegador modificado, é uma outra estratégia pró-activa. Estes agentes enviarão informação directamente a partir do navegador do utilizador para o servidor *Web*. Desta forma, é possível a obtenção de dados fidedignos das iterações do utilizador com o servidor *Web*. Em [Clickstream01] é proposta a utilização de um agente desenvolvido em *Java* para este efeito.

Embora referenciado como um elemento de distinção entre utilizadores, a utilização da autenticação poderá ser também entendida como parte de uma estratégia mista de identificação de sessão. É mista pois é efectuado pró-activamente a distinção entre os utilizadores mas a atribuição de um identificador de sessão único é feito *a posteriori*, ou seja reactivamente, aquando do processamento dos *logs*.

6.2.2 Estratégias Reactivas

Caso a atribuição do identificador de sessão não seja feita durante a ocorrência dos pedidos HTTP, quer automaticamente pelo servidor *Web* quer programaticamente pelas páginas do sítio *Web*, então este terá de ser atribuído posteriormente através do processamento dos ficheiros de *log*. O mesmo acontecerá para os pedidos executados por agentes HTTP que não aceitaram, por não poderem ou por não quererem, os identificadores de sessão que lhes foram atribuídos e transmitidos nas respostas aos seus pedidos. Neste caso a análise dos registos no *log* tem de determinar quais os pedidos efectuados pelo mesmo utilizador bem como identificar quando termina uma sessão e se inicia outra. Vejamos então algumas heurísticas (Tabela 6.3) que podem ser utilizadas para a reconstrução de sessões. Para tentar obter um valor do tempo total

despendido numa visita a um sítio *Web* Catledge e Pitkow [CatledgePitkow95] mediram o valor médio de inactividade dentro de um sítio e chegaram a um resultado final de 25.5 minutos. Vários sistemas, por exemplo o sistema descrito em [Cooley et al. 99], arredondaram este valor para 30 minutos e passaram a usar esse tempo como sendo o tempo máximo para a duração de uma sessão. A primeira heurística, **Ht1** é derivada deste estudo.

Uma refinação deste valor leva-nos à heurística **Ht2**. Reconhecendo que existem diversos tipos de páginas - raiz, de navegação, de conteúdo – poderá ser definido um tempo pré-determinado, em função do tipo de página, como sendo o tempo máximo de permanência numa página. Se o tempo entre duas páginas **p** e **q** servidas consecutivamente for superior ao pré-determinado para **p** então a página **q** seria incluída numa nova sessão. Está heurística é também usada em [Cooley et al. 99].

Os métodos utilizados para pré-processamento dos dados, neste caso a nível da reconstrução da sessão, deverão ser convenientemente seleccionados pois diferentes métodos podem causar diferenças substanciais nas conclusões que podem ser extraídas [Zheng et al. 03].

Heurística	***Descrição***
Ht1 - Tempo total de sessão	Assumir um tempo pré-determinado para a duração total da sessão, 30 minutos por exemplo.
Ht2 - Tempo por página	Assumir um tempo pré-determinado, 10 minutos por exemplo, ao fim do qual se considera que, se não houver registos no *log* para o mesmo utilizador então a sessão terminou.
Href - Baseado no referenciador	Sejam **p** e **q** duas páginas consecutivas com **p** pertencendo a uma sessão **S**. Considere-se $\mathbf{t_p}$ e $\mathbf{t_q}$ as estampilhas temporais das páginas **p** e **q** respectivamente. Então **q** pertence à sessão **S** se o referenciador de **q** foi previamente invocado em **S** ou, caso o referenciador de **q** for indefinido, $\mathbf{(t_q - t_p)} \leq \Delta$ para um intervalo de tempo Δ pré-determinado. Caso contrário **q** será adicionada a uma nova sessão.

Tabela 6.3 – Heurísticas para a reconstrução de sessões

Em [Spiliopoulou et al. 03] e [Berendt et al. 02] é analisado o impacto que a estrutura do sítio *Web*, nomeadamente a existência ou ausência de *frames* nas páginas servidas, pode ter na precisão destas heurísticas. As heurísticas a utilizar para a reconstrução das sessões deverão ser seleccionadas em função das características do sítio *Web*. Em [Berendt et al. 02] é indicado que as heurísticas **Ht1** e **Ht2** são apropriadas para sítios *Web* tanto com como sem *frames*. O desempenho da heurística **Href** é melhor em sítios sem *frames* e com sessões de pequena duração.

Enquanto que a identificação da primeira página numa sessão poderá ser relativamente simples, pelo início de registo de pedidos no ficheiro de *log*, a identificação da última página já não é uma tarefa tão trivial. Devido ao funcionamento em modo de pedido e resposta do protocolo http, não há nenhum registo nos *logs* do servidor *Web* a dizer que o utilizador terminou a sua visita. Esta saída pode acontecer porque o utilizador seleccionou um apontador que o levou a outro sítio *Web*, carregou no botão de *Home* do seu navegador, seleccionou um outro sítio *Web* da sua lista de favoritos, digitou outro URL ou simplesmente fechou o navegador. Com estas acções cessa por completo o registo de actividades desse utilizador nos *logs* do servidor *Web*. Mas estas podem ser apenas suposições já que o utilizador pode pura e simplesmente ter deixado de interagir com o sítio *Web* para atender uma chamada telefónica e ainda ter uma página do sítio no seu navegador. Se o utilizador voltar a interagir com o sítio *Web* então o registo de actividade no ficheiro de *log* é retomado. Existem, todavia, algumas optimizações que poderão ser feitas nas heurísticas apresentadas, mais especificamente nas temporais, que poderão ajudar a determinar o momento exacto do fim de uma sessão. Se o sítio *Web* forçar o utilizador a autenticar-se para que lhe seja concedido acesso aos conteúdos então deverá ser também disponibilizada uma opção de "Terminar", ou "Sair". Se o utilizador seleccionar esta opção então uma invocação a uma página específica poderá ficar registada no ficheiro de *log*. Este registo determina o momento exacto em que se pode dar por terminada uma sessão. Lembre-se, contudo, que a simples existência desta opção não é um garante que os utilizadores a vão utilizar. Estes podem simplesmente optar por fechar o navegador e, neste caso, nenhum evento de saída ficará registado. Uma outra optimização passa pela substituição dos apontadores para URLs em sítios externos por invocações a uma página de redireccionamento. No exemplo apresentado de seguida, vemos a possível instrução em html para invocação de uma página *asp* que teria o comando para efectuar o redireccionamento para o sítio externo mencionado no parâmetro `apt` que lhe é passado no URI:

```
<A href=”/redir.asp?apt=http://www.google.com”>Motor de
pesquisa</A>
```

Sempre que houver uma saída do sítio *Web* por um destes apontadores, a invocação da página `redir.asp`, e o seu parâmetro, fica registada no ficheiro de *log*. Sabemos assim quando e para onde houve uma saída. Este registo, por sua vez, poderá ser utilizado para indicar o ponto final de uma sessão. Este método poderá ser também aproveitado para fins comerciais já que, se for um apontador para um sítio que paga pelo tráfego recebido, permitirá contabilizar quantos acessos foram redireccionados a esse sítio.

6.2.3 Problemas com Tempos Inconsistentes

Cada entrada nos ficheiros de *log* é registada com uma etiqueta temporalque indica o data e hora do pedido HTTP. Se forem usados vários servidores *Web* então os seus ficheiros de *log* terão de ser agrupados num único onde todas as entradas estão ordenadas cronologicamente. Para que isto seja possível é necessário que a hora de cada um dos servidores *Web* esteja sincronizada até ao décimo de segundo [KimballMerz00]. Caso esta sincronização não exista, qualquer interpretação dos dados, no que diz respeito à sequência de eventos e comportamento dos utilizadores, será extremamente difícil. Vejamos um exemplo de um acesso a um sítio *Web* fictício que toma o nome `www.sitioweb.org` [Borges et al. 03]. Este sítio está alojado em vários servidores *Web* replicados. Suponhamos que a primeira linha é do ficheiro de *log* do servidor **C** e a segunda linha é do ficheiro de *log* do servidor **D**.

```
10.32.10.1 - - [26/Apr/2003:20:23:58 +0100] "GET /index.html
HTTP/1.1" 200 2123 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows
NT 5.0)"
```

```
10.32.10.1 - - [26/Apr/2003:20:24:05 +0100] "GET
/manual/chapt1.html HTTP/1.1" 200 2401
"http://www.asite.org/index.html" "Mozilla/4.0 (compatible; MSIE
6.0; Windows NT 5.0)"
```

A interpretação que pode ser feita a partir destes dois registo é que o utilizador acedeu ao sítio `www.sitioweb.org`, tanto pela selecção de um apontador da sua lista de favoritos ou por ter simplesmente digitado o endereço do sítio *Web* no seu navegador, e abriu a página `/index.html`. este utilizador sabia exactamente onde ir pois apenas 7 segundos depois ele seleccionou um apontador que o levou para a página `/manual/chapt1.html`.

Suponhamos agora que o relógio interno do servidor *Web* **D** estava atrasado em cerca de 10 segundos em relação ao relógio do servidor **C**. Com este desfasamento temporal, a operação de ordenação cronológica do ficheiro de *log* unificado fica alterada. O servidor **D** registou a primeira entrada enquanto que a segunda foi registada pelo servidor **C**.

```
10.32.10.1 - - [26/Apr/2003:20:23:55 +0100] "GET
/manual/chapt1.html HTTP/1.1" 200 2401
"http://www.asite.org/index.html" "Mozilla/4.0 (compatible; MSIE
6.0; Windows NT 5.0)"

10.32.10.1 - - [26/Apr/2003:20:23:58 +0100] "GET /index.html
HTTP/1.1" 200 2123 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows
NT 5.0)"
```

Estas entradas no *log* sugerem agora uma interpretação diferente. A primeira linha diz-nos que o utilizador apanhou a página `/index.html` em algumas das *cache*s existentes entre o seu computador e o servidor *Web*, por exemplo no seu próprio navegador. Chegou até à página `/manual/chapt1.html` após ter clicado num apontador contido na página servida a partir da *cache*. A segunda linha do *log* poderá indicar que o utilizador digitou o endereço URL da página `/index.html` directamente numa nova janela do seu navegador. Ora isto não faz sentido já que a partir da interpretação da primeira linha do *log* inferimos que o utilizador tinha esta página em *cache* . Nesse caso, esta segunda linha não deveria existir no *log* ou, então, deveria ter um código de estado HTTP igual a `304`, significando "Não Modificado", em vez de `200` que tem o significando de "OK". Pela análise destas duas entradas não é possível chegar a uma conclusão sobre qual terá sido, neste caso, o real comportamento do utilizador.

Quando estivermos na presença de um sítio *Web* replicado por vários servidores na mesma rede local, uma forma que temos para evitar este tipo de problemas será a de sincronizar o relógio de todos os servidores recorrendo ao *Network Time Protocol* (NTP) [Lombardi99][Mills92]. Todos os servidores *Web* podem assim obter e observar a data e hora mantida pelo servidor NTP. À falta de um servidor NTP interno poderia ser utilizado, em Portugal, o servidor NTP do Observatório Astronómico de Lisboa. Esta instituição tem a responsabilidade de manter a hora legal portuguesa e mantém publicamente acessível essa hora através do servidor `ntp02.oal.ul.pt`.

Podemos também encontrar situações onde uma organização tenha a sua presença na *Web* disseminada por vários servidores em localizações distintas, e eventualmente em diferentes fusos horários. Devido à latência, por vezes imprevisível, que é possível encontrar entre redes alargadas (WANs), ou ligações através da Internet, é recomendável a existência de um servidor NTP em cada localização. Estes servidores NTP deverão, por sua vez, ser sincronizados com um *standard* que seja usado por toda a organização. Este *standard* deverá ser, preferivelmente, o *Greenwich Mean Time* (GMT) ou *Coordinated Universal Time* (UTC) já que os seus valores podem ser obtidos tanto por rádio como por telefone [Lombardi01] ou ainda a partir dos satélites de posicionamento global (GPS) [Lombardi02]. Convém relembrar que alguns servidores *Web* usam a hora local por omissão. Neste caso, a sua configuração deverá ser alterada para passarem a utilizar a hora GMT ou UTC.

Quando se estiver a interpretar a informação da etiqueta temporalregistada nos ficheiros de *log* deve ser tida em conta a forma como diferentes servidores *Web* lidam com a chamada "hora de verão". Foram feitos testes comparativos no mesmo computador, com o Microsoft Windows 2000 Professional instalado, correndo ora o Microsoft IIS v5 ora o *Apache* v1.3, ambos configurados para registarem os acessos ao sítio *Web* em *logs* com o *NCSA Common Log Format*, e diferentes resultados foram obtidos [Borges et al. 03]. Com o computador configurado para não efectuar a mudança da hora automaticamente e o fuso horário indicando *GMT Standard*, o servidor *Web Apache* registou a seguinte entrada no ficheiro de *log*:

```
10.32.10.1 - - [04/May/2003:00:08:55 +0000] "GET /inicio.htm
HTTP/1.1" 200 798
```

Alterou-se o servidor *Web* para o Microsoft IIS e, cerca de dois minutos depois, o acesso à mesma página resultou na seguinte entrada no ficheiro de *log*:

```
10.32.10.1 - - [04/May/2003:00:10:20 +0000] "GET /inicio.htm
HTTP/1.1" 200 1068
```

Ou seja, não se registou nenhuma diferença para além da hora de acesso. Quando o computador foi configurado para efectuar a mudança da hora automaticamente e tendo o fuso horário indicando "GMT Dayligh Time" os resultados foram diferentes. Com o servidor *Apache* a correr o registado do acesso gerou no ficheiro de *log* a seguinte entrada:

```
10.32.10.1 - - [04/May/2003:00:55:38 +0100] "GET /inicio.htm
HTTP/1.1" 200 798
```

O mesmo acesso já com o Microsoft IIS a correr gerou o seguinte registo:

```
10.32.10.1 - - [04/May/2003:00:57:08 +0000] "GET /inicio.htm
HTTP/1.1" 200 1022
```

Fazendo uma análise a este caso nota-se uma clara diferença na forma como os dois servidores *Web* registam o desfasamento temporal em relação à hora GMT. No caso do *Apache* é de `+0100` e no caso do Microsoft IIS é `+0000`. Se, eventualmente, houver a necessidade de juntar os *logs* no formato da NCSA gerados por estes dois servidores *Web* então esta inconsistência de representação deverá ser tida em consideração.

Mesmo que uma organização sincronize os relógios de todos os seus servidores *Web* isso não é uma garantia que este tipo de inconsistências temporais não ocorram se usar dados de *clickstream* provenientes de entidades externas. A organização deverá garantir que estes dados externos foram gerados por sistemas também eles sincronizados com o mesmo *standard* temporal usado internamente. Assim sendo, evitar-se-ão as inconsistências temporais devido a ter servidores *Web* a trabalhar com hora local e outros com hora GMT ou UTC para além dos eventuais desfasamentos dos relógios dos servidores.

6.3 Identificação de Páginas

Todos os pedidos registados nos *logs* dos servidores *Web* tem de ser analisados por forma a poder classificar os objectos servidos como sendo páginas *Web* ou simplesmente componentes de uma página. Caso esses objectos sejam elementos componentes de uma página *Web* então deverá ser identificado que página é essa. A identificação de que página fazem parte será determinada pelo referenciador indicado no registo do *log*. Esta identificação pode ainda ser, eventualmente, complementada com informação constante no Descritor de Estrutura e Conteúdo do sítio *Web*.

As páginas visitadas deverão ser identificadas e a sua ocorrência interpretada no contexto de uma sessão. Cada página deverá ser univocamente identificada através de um número de sequência dentro da sessão. Esta sequencialização facilitará o cálculo das diversas medidas constantes nas tabelas de factos. Um factor que deve ser tido em conta na identificação e sequencialização das páginas no contexto da sessão é o efeito que as *caches* causam. A utilização de *caches* é sem dúvida um excelente mecanismo de melhorar tempos de resposta e diminuir o tráfego na rede. No entanto, a sua utilização traz alguns problemas do ponto de vista da recolha de dados em *logs* dos servidores *Web*. Os navegadores dos utilizadores e os diversos *proxies* de *cache,* existentes entre a localização do utilizador e o servidor *Web,* guardam localmente cópias das páginas que foram acedidas. Se um utilizador usar o botão de retrocesso, ou avanço*,* do seu navegador, este poderá aceder às páginas que estejam guardadas localmente em vez de fazer um novo pedido ao servidor. De igual forma, poderá haver um *proxy* de *cache* no ISP do utilizador que já tenha essa página armazenada. Essa página será assim servida pelo ISP localmente da sua *cache* e poderá não haver o correspondente pedido HTTP enviado ao servidor *Web*. Consequentemente, esse acesso nunca ficará registado no *log* do servidor. As *caches* são uma das razões porque muitos analisadores de *logs* reportam que as páginas mais populares são páginas situadas mais abaixo na estrutura do sítio. As primeiras páginas já estarão certamente em *cache* algures na Internet. Existem, no entanto, alguns métodos que podem ser utilizados (Tabela 6.4) como forma de reduzir a falta de registos nos ficheiros de *log* de páginas servidas a partir de *cache*.

Método	***Vantagens***	***Desvantagens***
Utilização de páginas dinâmicas	Reduz o número de acessos "perdidos".	Aumenta o tráfego no servidor *Web*.
Utilização de um componente dinâmico na página (ex: imagem de um pixel de cor transparente e com um nome, ou parâmetro no URI, aleatório)	Pela análise do referenciador dos pedidos desta imagem todas as páginas visitadas pelo utilizador são identificadas.	Não temos acesso a informação que estaria presente no pedido da página original se esta for servida de *cache* (ex*: referenciador da página original*). Implica a modificação de todas as páginas do sítio *Web*.
Atribuir às páginas Web datas de validade no passado para forçar os navegadores a pedirem uma nova versão sempre que são acedidas	Pode ser efectuado selectivamente. Reduz o número de acessos "perdidos".	Aumenta o tráfego no servidor *Web*. *Proxies* e navegadores podem não respeitar esta indicação.
Utilizar directivas no cabeçalho do código html que indiquem que a página não deve ser mantida em cache	Pode ser efectuado selectivamente. Reduz o número de acessos "perdidos".	Aumenta o tráfego no servidor *Web*. *Proxies* e navegadores podem não respeitar esta indicação.

Tabela 6.4 – Métodos para reduzir problemas originados por *Cacheing*

Caso não seja utilizado nenhum método para prevenir o efeito das *caches* no registo em *log* das páginas servidas então os registos omissos poderão, eventualmente, ser inferidos. Após a ordenação pela etiqueta temporaldas páginas servidas na sessão podemos analisar o valor dos seus referenciadores. Se o referenciador de uma página **P** não for igual à página que a precede cronologicamente na mesma sessão então podemos inferir que houve uma, ou mais, páginas que foram servidas a partir de uma *cache* até ao pedido dessa página **P**. Nesse caso, se a página referenciadora de **P** tiver sido previamente invocada na sessão do utilizador então podemos assumir que o utilizador usou o botão de retrocesso do seu navegador, visualizando páginas armazenadas na sua *cache*, até a página **P** ter sido pedida [Cooley et al. 99]. Vejamos um exemplo para nos ajudar a perceber como poderíamos completar o caminho percorrido pelo utilizador a partir dos acessos registados no *log*. No exemplo ilustrado (Figura 6.1) [Mobasher et

al. 01] podemos ver a representação da estrutura de um sítio *Web* com indicação dos apontadores existentes entre as páginas *Web* e, a vermelho (ou na cor mais escura em impressões monocromáticas), temos a indicação da sequência das páginas visitadas por um utilizador.

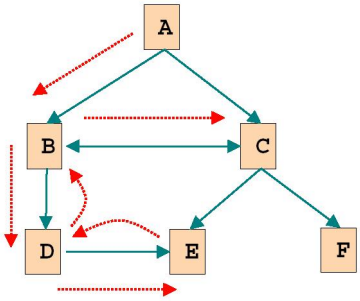

Figura 6.1 – Estrutura do sítio *Web* e páginas visitadas

Se no ficheiro de *log* tivéssemos algo do género (para simplicidade não são mostrados todos os campos do *log*):

```
     Data e Hora              URL         Referenciador
[04/May/2003:11:10:20 +0000] A.html      -
[04/May/2003:11:10:30 +0000] B.html      A.html
[04/May/2003:11:10:35 +0000] D.html      B.html
[04/May/2003:11:10:40 +0000] E.html      D.html
[04/May/2003:11:10:55 +0000] C.html      B.html
```

Analisando o exemplo vemos que para a última página servida, `C.html`, o referenciador toma o valor de `B.html`. Ora essa página não corresponde à página que a precede cronologicamente no *log*, `E.html`. Como a página `B.html` foi previamente invocada então podemos inferir que o utilizador utilizou a tecla de retrocesso do seu navegador antes de ter chegado à página `C.html`. Considerando a estrutura do sítio *Web* e o valor do referenciador de `C.html` dois caminhos podem ser sugeridos:

```
A.html -> B.hmtl -> D.html -> E.html -> D.html -> B.html -> A.html
-> B.html -> C.html
```

Ou então:

```
A.html -> B.hmtl -> D.html -> E.html -> D.html -> B.html -> C.html
```

Nestes casos em [Cooley et al. 99] é sugerida a heurística de usar sempre o caminho mais curto. Sendo assim chega-se ao caminho:

```
A.html -> B.hmtl -> D.html -> E.html -> D.html -> B.html -> C.html
```

Seriam então incluídas na sessão as duas páginas em falta no ficheiro de *log*:

```
     Data e Hora              URL         Referenciador
[04/May/2003:11:10:20 +0000] A.html      -
[04/May/2003:11:10:30 +0000] B.html      A.html
[04/May/2003:11:10:35 +0000] D.html      B.html
[04/May/2003:11:10:40 +0000] E.html      D.html
[??????????????????????????] D.html      E.html
[??????????????????????????] B.html      D.html
[04/May/2003:11:10:55 +0000] C.html      B.html
```

Será também necessário um algoritmo para determinar a etiqueta temporaldas páginas agora incluídas. Uma possibilidade será o uso do tempo médio de permanência em páginas do mesmo tipo. Esse tempo médio pode ser usado para subtrair ao tempo do pedido da página `C.html` para obtenção do tempo do pedido da página `B.html` e posteriormente do tempo de `D.html`. Supondo esse tempo médio igual a 5 segundos então teríamos o seguinte resultado final:

```
     Data e Hora              URL         Referenciador
[04/May/2003:11:10:20 +0000] A.html      -
[04/May/2003:11:10:30 +0000] B.html      A.html
[04/May/2003:11:10:35 +0000] D.html      B.html
[04/May/2003:11:10:40 +0000] E.html      D.html
[04/May/2003:11:10:45 +0000] D.html      E.html
[04/May/2003:11:10:50 +0000] B.html      D.html
[04/May/2003:11:10:55 +0000] C.html      B.html
```

Como vimos pelo exemplo é de extrema importância a existência do `Descritor de Estrutura e Conteúdo do Sítio Web`. Enquanto que para sítios *Web* estáticos este seja relativamente simples de manter pelos programadores em sítios dinâmicos será necessário, pelo menos, a existência de uma ferramenta de mapeamento que através de pedidos HTTP sucessivos ao sítio *Web*, ou outro método, consiga obter informação sobre a estrutura do mesmo.

6.4 Construção das Dimensões e Tabela de Factos

A identificação de utilizadores, reconstrução de sessões e identificação de páginas são um pré-processamento necessário dos dados de *clickstream*. Este pré-processamento facilitará a construção das várias dimensões e tabelas de factos. Quando terminado, ficaremos com os utilizadores diferenciados e classificados como agentes automáticos ou humanos, as sessões univocamente identificadas, os objectos *Web* devidamente sequenciados e as entradas omissas no *log* devidamente corrigidas. Após estes passos, efectua-se o processamento das dimensões, com as respectivas chaves de substituição, e construção das tabelas de factos.

As dimensões do *Data Webhouse* podem ser variadas e dependem tanto dos dados e sistemas existentes como dos objectivos e análises pretendidas. As dimensões apresentadas no capítulo 4 são vocacionadas para análises em sítios *Web* transaccionais. A sua construção será aqui alvo de uma descrição de alto nível tocando o tema da origem dos seus dados e processo de gestão de alterações dos dados já quando inseridos nas dimensões. Como a estrutura de algumas das dimensões depende em grande parte dos sistemas existentes em cada organização, não é possível ter uma descrição mais aprofundada sem conhecimento destes.

A dimensão `Sítio Web` pode ser construída a partir de uma tabela de configuração que possa existir na Zona de Concentração de Dados. Esta tabela seria construída e mantida manualmente e, tipicamente, os seus conteúdos seriam de cariz bastante estático.

Tanto a dimensão `Data` como a dimensão `Tempo` podem ser geradas de forma automatizada. A dimensão `Tempo` pode mesmo ser criada no momento em a *Data Webhouse* é criada. Esta é uma

dimensão com conteúdo estático. Quanto à dimensão `Data` também pode ser gerada no momento da criação do *Data Webhouse* para o período temporal que se queira inicialmente abranger. Contudo, novos valores terão de ser adicionados à dimensão quando os existentes deixarem de ser suficientes para as análises pretendidas.

A dimensão do `Método HTTP` é criada e mantida manualmente, a partir das definições do protocolo HTTP, numa tabela na Zona de Concentração de Dados e depois replicada para o *Data Webhouse*. Esta tabela tem conteúdo bastante estático e apenas será alterada se forem efectuadas alterações ao protocolo HTTP. Poderá mesmo ser considerada a hipótese de a manter exclusivamente no *Data Webhouse*.

No caso da dimensão `Agente HTTP` os seus valores são resultado do processamento dos registos existentes nos ficheiros de *log*. O processo de identificação do utilizador contribui, neste caso, para determinar qual o tipo de agente e distingir se este é um navegador ou um agente automático.

A construção e manutenção da dimensão `Estado HTTP` é idêntica à dimensão `Método HTTP`. É mantida manualmente na ZCD e posteriormente replicada para o *Data Webhouse*. Após a inserção inicial de dados apenas sofrerá alterações se o protocolo HTTP for alterado.

Relativamente à dimensão `Computador do Utilizador` esta será construída a partir do processamento dos dados constantes nos ficheiros de *log* do servidor *Web*. Será também complementada com informação que terá de ser obtida a partir de servidores de `DNS` bem como servidores que mantenham a base de dados `Whois` específica para o endereço IP fornecido. Como diferente informação poderá estar associada a um endereço IP, esta dimensão será do tipo 2 segundo a classificação dada em [Kimball et al. 98], ou seja, se informação associada a um endereço IP for diferente da já existente então um novo registo será inserido na dimensão.

A dimensão `Utilizador` é um conglomerado de informação proveniente de distintos sistemas. A construção desta dimensão procede o trabalho iniciado pela distinção entre utilizadores previamente efectuada. Um registo nesta dimensão na sua forma mais incompleta poderá simplesmente prover dos ficheiros de *log*. Este seria o caso de um utilizador anónimo. Em situações de utilizadores registados teríamos informação proveniente dos ficheiros de *log*, do

sistema de controlo de acessos, e dos sistemas operacionais e aplicações *Web* necessárias para manter o ambiente transaccional. Os registos desta dimensão poderão ser alterados em função da informação que se vá obtendo progressivamente sobre os utilizadores. As alterações efectuadas a esta dimensão serão do tipo 1 segundo a classificação dada em [Kimball et al. 98], ou seja, serão feitas no mesmo registo rescrevendo os antigos valores. Caso haja interesse em manter uma partição temporal da informação associada a um mesmo utilizador então a estrutura da dimensão deverá ser alterada para contemplar um campo com o a data da última alteração e outro com o descritivo da alteração efectuada.

A informação constante na dimensão `Entidade` é tipicamente originária dos sistemas operacionais de suporte às transacções comerciais. Informação sobre clientes e fornecedores estará aqui contida. Poderá, contudo, ser também acrescida manualmente com informação sobre outras entidades não existentes nesses sistemas. Este é o caso de, por exemplo, informação sobre os diversos motores de pesquisa que sejam considerados relevantes para as análises no *Data Webhouse*. Estes casos de informação não proveniente dos sistemas operacionais terão de ser criados e mantidos em tabelas auxiliares na ZCD. As alterações de dados nesta dimensão são efectuadas sobre os próprios registos. A tabela auxiliar desta dimensão na *Webhouse*, `Perfil de Entidade`, terá informação que será preenchida tanto automaticamente, se dados da entidade a que se reporta derivarem dos sistemas operacionais, como manualmente para os casos da informação sobre as entidades provirem da tabela mantida na ZCD. Um novo registo é criado sempre que se verifiquem alterações aos dados relativos ao perfil da entidade.

Os valores constantes na dimensão `Referenciador` são provenientes dos *logs* do servidor *Web*, de servidores de `DNS` bem como servidores que mantenham a base de dados `Whois` específica para o endereço IP fornecido. A informação dos *logs* também é cruzada com informação que pode provir dos sistemas operacionais para identificação das entidades referenciadoras ou da tabela com informação sobre outras entidades que existirá na ZCD. As alterações a dados existentes são efectuadas pela adição de um novo registo com respectiva indicação de data de alteração.

Os dados inseridos na dimensão `URI` provêm apenas dos ficheiros de *log* do servidor *Web* eventualmente complementada com informações de URIs que estavam omissos do *log* mas foram

integrados aquando do pré-processamento. Nesta dimensão apenas serão inseridos novos registos, não se efectuando alterações aos registos existentes.

A dimensão `Objecto Web` obtém os seus dados essencialmente a partir de duas fontes. Do `Descritor de Estrutura e Conteúdo do Sítio Web` e dos *logs* do servidor *Web*. Em situações de sítios *Web* com páginas geradas dinamicamente, a informação poderá ser complementada pelos dados fornecidos por ferramentas de mapeamento da estrutura e conteúdo do sítio *Web*. Esta dimensão mantém um histórico das alterações efectuadas aos objectos *Web*. Se, por exemplo, uma nova versão de uma página for criada por um programador, então será criado um registo novo a partir dos dados do registo existente para esse objecto mas reflectindo as alterações efectuadas.

A dimensão `Produto`**,** na forma apresentada**,** é alimentada com dados provenientes dos sistemas operacionais de suporte ao negócio. Se maior detalhe for necessário nesta dimensão no que diz respeito, por exemplo, à categorização dos produtos, então poderá haver a necessidade de a criar e manter em tabelas auxiliares na ZCD. Igual procedimento seria necessário se esta dimensão fosse de Serviços e os mesmos não fossem completamente descritos nos sistemas operacionais. Note-se que a estrutura apresentada desta dimensão não contempla manter um histórico das alterações. Se estas ocorrerem, então os registos já existentes na dimensão serão rescritos. Este comportamento é classificado como sendo do tipo 1 por [Kimball et al. 98] no que diz respeito à política de alterações da dimensão.

Quando chegamos à dimensão `URI Página` podemos pensar nela como uma selecção a partir da dimensão `URI` onde apenas os URIs que identificam uma página são escolhidos. Nesse caso, as fontes originais de dados serão as mesmas que foram necessárias para a construção da dimensão `URI`. Todas as alterações a esta dimensão serão reflectidas pelo acréscimo de um novo registo. Registos com os anteriores valores são mantidos como histórico.

A informação para a dimensão `Promoção` será essencialmente mantida manualmente em tabela auxiliar existente na ZCD. Isto para o caso das aplicações *Web* ou os sistemas operacionais não terem a capacidade de manter informação relativas a promoções. Tal como na dimensão Página, alterações a valores nesta dimensão traduzir-se-ão pelo acréscimo de novos registos.

O método de obtenção de informação para as dimensões `Cesto de Compras` e `Actividade` dependerá totalmente de como o sistema transaccional está implementado no sítio *Web*. O conceito do cesto de compras muito provavelmente só existirá dentro das aplicações *Web*. Depreende-se que serão estas que fornecerão os seus dados. Relativamente às actividades registadas sobre o cesto de compras estas poderão ser indicadas por algum URI específico registado nos ficheiros de *log* ou então através de informação registada na base de dados das aplicações *Web*. Se pensarmos na dimensão `Actividade` como sendo de cariz estático, onde todos os possíveis valores são conhecidos à partida, então esta poderá ser carregada programaticamente, ou mesmo manualmente, aquando da criação da *Data Webhouse*.

Uma vez que todas as dimensões contenham os seus valores finais, e sem registos duplicados, podem ser então atribuídas as chaves de substituição. Tipicamente as chaves de substituição tomam valores numéricos sequenciais e desprovidas de qualquer valor semântico. Serão estas chaves de substituição que serão usadas, como chaves estrangeiras, nas tabelas de factos e permitir uma ligação com as diversas dimensões. Deverão existir na ZCD tabelas que permitam associar os valores das chaves usadas nos sistemas operacionais, onde aplicável, com as chaves de substituição. Estas tabelas, ao contrário de outras tabelas auxiliares do processo de ETI, deverão permanecer entre operações de carga. Em [Kimball et al. 98] pode ser consultado um método de atribuição deste tipo de chaves às tabelas dimensionais.

Após a criação das dimensões, e respectivas chaves de substituição, deverão ser construídas as tabelas de factos com o grau previamente escolhido. No capítulo 4 foram descritas três tabelas de factos:

- Tabela de Factos para Análise de Pedidos HTTP.
- Tabela de Factos para Análise de Páginas *Web*.
- Tabela de Factos para Análise de Sessões Completas.

São várias as medidas existentes na `Tabela de Factos para Análise de Pedidos HTTP` (Tabela 6.5). Todas estas medidas poderão ser calculadas com dados extraídos directamente dos ficheiros de *log* dos servidores *Web* e, eventualmente, complementados com informação proveniente de ficheiros de *log* de servidores *proxy* de *cache*.

Medida	*Descrição*	*Fórmula de Cálculo*	*Regra de agregação*
Bytes recebidos	Número de *bytes* recebidos pelo servidor *Web* num pedido HTTP. Este valor não inclui o tamanho dos cabeçalhos do protocolo HTTP.	Extraído directamente do valor registado nos ficheiros de *log*.	Soma.
Bytes enviados	Número de *bytes* enviados pelo servidor *Web* ao cliente HTTP como resposta ao pedido HTTP. Este valor não inclui o tamanho dos cabeçalhos do protocolo HTTP.	Extraído directamente do valor registado nos ficheiros de *log.*	Soma.
Segundos para servir	Número total de segundos que um objecto demorou a ser servido incluindo o carregamento das imagens ou animações que dela fazem parte. Não incluí tempo de transmissão na rede.	Extraído directamente do valor registado nos ficheiros de *log*. Se objecto adicionado durante o pré-processamento então valor é extraído do Descritor de Estrutura e Conteúdo.	Soma.
É Encriptada	Valor que indica se objecto foi transmitido encriptado para o cliente HTTP.	Se valor do protocolo registado no ficheiro de *log* for https então toma o valor de "Sim" caso contrário toma o valor de "Não". Se objecto ou página for inferido então toma o valor de "Não".	Não aplicável.
É Encriptada	Valor que indica se objecto foi transmitido encriptado para o cliente HTTP.	Se valor do protocolo registado no ficheiro de *log* for https então toma o valor de "Sim" caso contrário toma o valor de "Não". Se objecto ou página for inferido então toma o valor de "Não".	Não aplicável.

Tabela 6.5 – Factos medidos da Tabela de factos para análise de pedidos http

Se às entradas registadas nos ficheiros de *log* do servidor *Web* forem adicionadas referências a páginas, ou objectos, *Web* omissas dos *logs* durante o pré-processamento, induzidas pela existência de *caches*, então os valores das medidas `Bytes Recebidos`, `Bytes Enviados` e `Segundos Para Servir` terão de ser estimados. No cálculo desta estimativa podem ser adoptadas duas estratégias:

- Considerar que estas medidas tomam o valor zero, já que os respectivos pedidos não passaram realmente pelo servidor *Web* por terem sido servidos a partir de uma *cache* na rede.
- Usar valores registados para os mesmos URIs em pedidos anteriores ou, se estes não existirem, o cálculo a partir de valores médios para objectos de idênticas características.

Seja qual for a estratégia adoptada, esta terá de ser consistentemente considerada ao longo dos restantes cálculos e análises a serem efectuadas.

Para a `Tabela de Factos para Análise de Páginas` *Web* é importante a definição do conceito de página pois é este que determinará qual o tipo de pedidos HTTP sobre o qual esta tabela reportará os factos medidos. Este é um conceito que pode variar por organização e se, por exemplo, uma imagem ou ficheiro vídeo pode ter significado para análise como "página" numa organização, numa outra apenas os ficheiros html poderão ser considerados como páginas.

A forma de cálculo das medidas desta tabela de factos (Tabela 6.6) `Bytes Página Recebidos`, `Segundos Entre Páginas`, `Segundos Para Servir Página` e `Segundos Visualização Página` variam todas também em função do seu interesse para a análise final. Aqui poderá, ou não, haver interesse em incluir todos os objectos utilizados para compor a página *Web*, tal como ela é apresentada ao utilizador final, ou apenas no próprio ficheiro da página. Se este tipo de rigor técnico for dispensável então apenas os dados específicos ao pedido HTTP do URI da página deverão ser utilizados.

Medida	Descrição	Fórmula de Cálculo	Regra de Agregação
Bytes página recebidos	Número de *bytes* recebidos pelo sítio *Web* do cliente HTTP ao efectuar o pedido da página completa ao servidor. Este valor não inclui o tamanho dos cabeçalhos HTTP.	Este valor resulta da soma do número de *bytes* recebidos no pedido HTTP da página, registado nos ficheiros de *log*, bem como dos pedidos aos objectos que a constituem.	Soma.
Bytes página enviados	Número de *bytes* enviados pelo servidor *Web* ao cliente HTTP como resposta ao pedido da página. Este valor não inclui o tamanho dos cabeçalhos do protocolo HTTP.	Este valor resulta da soma do número de *bytes* enviados como resposta aos pedidos HTTP da página, registado nos ficheiros de *log*, bem como dos objectos que a constituem.	Soma.
Segundos entre páginas	Intervalo de tempo em segundos entre o pedido de uma página e o pedido da página seguinte.	Esta informação pode ser obtida a partir dos *logs* de *clickstream* e eventualmente complementada com informações extraídas do navegador do utilizador.	Soma.
Segundos para servir página	Número total de segundos que uma página demorou a ser servida incluindo o carregamento dos objectos que dela fazem parte. Não inclui tempo de transmissão na rede.	Soma dos tempos que tardou a servir a página mais cada um dos seus componentes. No caso da última página este valor poderá ter de ser um valor estimado.	Soma.
Segundos visualização página	Tempo estimado, em segundos, que a página esteve visível no navegador do utilizador. Não considera tempos de transmissão.	Calculado pela diferença entre medida `Segundos Entre Páginas` e medida `Segundos Para Servir Página`.	Soma.
Número de sequência na sessão	Valor numérico que indica, no contexto de uma sessão, em que posição foi a página servida ao utilizador. A primeira página servida ao utilizador na	Calculado pela análise da etiqueta temporaldas entradas registadas no ficheiro de *log* com o eventual complemento das páginas adicionadas aquando do pré-	Não aplicável.

Medida	Descrição	Fórmula de Cálculo	Regra de Agregação
	sessão terá sempre o número 1 a segunda página servida terá o número 2 e por aí adiante.	processamento.	
É última página	Valor que indica se página servida é a última no contexto de uma sessão.	Toma o valor de "Sim" ou "Não" e é determinado pela análise de todas as estampilhas temporais dos pedidos HTTP registados nos *logs*.	Não aplicável.
Unidades produto movimentadas para/de cesto	Número de unidades do produto que são adicionadas, ou removidas, ao cesto de compras.	Esta medida apenas terá valores diferentes de zero quando a actividade na página for Adicionar ao Cesto ou Remover do Cesto. Forma de cálculo, no que toca às fontes de dados, varia em função dos sistemas existentes.	Soma.
Valor produto movimentado para/de cesto em Euros	Valor do produto adicionado, ou removido, ao cesto de compras.	Calculado multiplicando o Unidades Produto Movimentadas para/de Cesto pelo preço unitário do produto movimentado. Esta informação pode ser obtida a partir das aplicações *Web* que dão suporte à transacção comercial.	Soma.
Valor da encomenda em Euros	Valor numérico que representa o valor em Euros de uma compra quando a actividade registada na página for Submeter Encomenda na maior parte dos registos tomará o valor zero.	Forma de cálculo varia em função da regras de negócio existentes na organização. Eventualmente algo do género: Número de unidades do produto multiplicada pelo seu preço unitário mais encargos de transporte menos descontos mais impostos.	Soma.

Tabela 6.6 – Factos medidos da Tabela de factos para análise de utilização de páginas *Web*

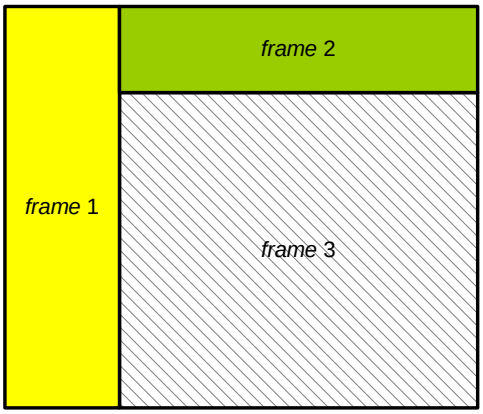

Figura 6.2 – Esquema de utilização de *frames*

O cálculo da medida `Segundos Entre Páginas` terá de ser diferente para a última página pois não existe, na mesma sessão, uma próxima página servida ao mesmo utilizador. Neste caso, a medida tomará o valor zero. O cálculo da medida `Número de Sequência na Sessão` só será possível após a sequencialização das páginas servidas. Tanto a medida `Segundos Entre Páginas` como a do `Número de Sequência na Sessão` poderão não fazer sentido caso o sítio *Web* faça um uso exaustivo de páginas em *frames* já que vários ficheiros que possam cair sob o conceito de página *Web* estarão simultaneamente a ser exibidas no navegador do visitante. A composição da página tal como ela é visível para o utilizador final implica, no exemplo apresentado (Figura 6.2), o pedido ao servidor *Web* quase que simultaneamente de três ficheiros distintos, uma para cada um dos *frames*. Também devido à forma como os pedidos HTTP são transmitidos na Internet não existe uma garantia absoluta que a sequência dos pedidos emitidos pelo navegador do utilizador para os diversos ficheiros de página usados nos *frames* chegue pela mesma ordem ao servidor *Web*. Os valores da medida `Número de Sequência na Sessão` deixaria de fazer sentido e os valores da medida `Segundos Entre Páginas` apresentariam um valor muito baixo, ou mesmo o valor zero caso os diversos ficheiros tivessem sido servidos no mesmo segundo.

As medidas de cariz comercial apenas farão sentido incluir para sítios *Web* transaccionais. O seu cálculo, no que diz respeito à fonte de dados, dependerá totalmente de como estão instalados os sistemas que dão suporte à transacção comercial complementada com indicador de submissão de

encomenda que estará ou registado nos *logs* de *clickstream* ou na base de dados das aplicações *Web*.

Para a `Tabela de Factos para Análise de Sessões Completas` as medidas apresentadas (Tabela 6.7) são, em parte, uma agregação das medidas já apresentadas para as tabelas de factos anteriores. A medida `Bytes Sessão Recebidos` pode ser obtida pela soma de todos os `Bytes Página Recebidos`, existente ao nível da `Tabela de factos para análise de utilização de páginas Web`. De igual forma, a medida `Bytes Sessão Enviados` pode ser calculada pela soma dos `Bytes Página Enviados` existente ao nível da página *Web*. Os `Segundos Para Servir Sessão` podem ser calculados ou pela soma de todos os `Segundos Entre Páginas`, existente ao nível da página, ou pela diferença entre a data e hora em que foi servida a primeira e última página mais o tempo de inactividade, depois de servida a última página, a partir do qual se considera que terminou uma sessão.

A medida `Segundos de visualização sessão` contém um valor que é calculado a partir do valor de duas outras medidas: `Segundos Duração da Sessão` menos o valor de `Segundos Para Servir Sessão`. Poder-se-ia optar por não incluir esta medida já que ela podia ser calculada directamente nas interrogações ao *Data Webhouse*. Essa opção será válida se a sua consulta for infrequente. Todavia, se o valor desta medida for frequentemente consultado então é preferível estar na tabela de factos já que tendo o seu valor pré-computado aumenta o desempenho das interrogações. A medida `Número de páginas Visitadas` poderá ter um valor totalmente deturpado se não houver o cuidado de preparar o sítio *Web* para anular o efeito das *cache*s e permitir registo de todas as páginas servidas.

A informação relativa ao `Número de Encomendas Submetidas` fará sentido, mais uma vez, apenas num contexto comercial. Esta informação provém das aplicações que dão suporte ao sistema transaccional do sítio *Web* conjugado ou pelos *logs* de *clickstream* se o sítio *Web* registar essa actividade com a chamada a um URI específico. Tal como a medida anterior, a medida `Número de Produtos Encomendados` existirá provavelmente apenas em sítios *Web* comerciais. O cálculo da medida `Valor das Encomenda Submetidas em Euros`, será uma soma do valor da sua correspondente existente a nível da página.

Medida	Descrição	Fórmula de Cálculo	Regra de Agregação
Bytes sessão recebidos	Número de *bytes* recebidos pelo sítio *Web* ao longo da duração da sessão. Este valor não inclui o tamanho dos cabeçalhos HTTP.	Este valor resulta da soma do número de *bytes* recebidos pelo servidor *Web* durante os pedidos HTTP efectuados ao longo da sessão.	Soma.
Bytes sessão enviados	Número de *bytes* enviados pelo sítio *Web* ao longo da duração da sessão. Este valor não inclui o tamanho dos cabeçalhos HTTP.	Este valor resulta da soma do número de *bytes* enviados pelo servidor *Web* durante os pedidos HTTP efectuados ao longo da sessão.	Soma.
Segundos de duração da sessão	Tempo total, em segundos, de duração de uma sessão.	calculado ou pela soma de todos os `Segundos Entre Páginas`, existente ao nível da página, ou pela diferença entre a data e hora em que foi servida a primeira e última página mais o tempo de inactividade, depois de servida a última página, a partir do qual se considera que terminou uma sessão.	Soma.
Segundos para servir sessão	Número total de segundos que tardou a servir todos os objectos numa sessão. Não inclui tempo de transmissão na rede.	resulta da soma do tempo tardado a servir todas as páginas *Web* e os seus componentes.	Soma.
Segundos de visualização sessão	Tempo estimado, em segundos, que as páginas *Web* servidas na sessão estiveram visíveis no navegador do utilizador. Não considera tempos de transmissão.	valor em segundos igual ao valor de `Segundos Duração da Sessão` menos o valor de `Segundos Para Servir Sessão`.	Soma.
Número de páginas	Indica quantas páginas foram servidas ao utilizador.	É simplesmente obtido pela contagem das páginas *Web* que	Soma.

Medida	Descrição	Fórmula de Cálculo	Regra de Agregação
Visitadas		foram servidas durante a sessão. Se a mesma página foi visitada duas vezes então contará como duas visitas.	
Número de Encomendas Submetidas	Este valor diz-nos quantas encomendas foram submetidas pelo utilizador durante a duração da sessão.	O seu valor é determinado pela contagem do número de vezes foi registada a actividade "Submeter Encomenda" durante a sessão.	Soma.
Número de Produtos Encomendados	Número de distintos produtos encomendados durante uma sessão.	Soma do número de distintos produtos registados em cada uma das encomendas submetidas durante a sessão.	Soma.
Valor das Encomenda Submetidas em Euros	Valor total das encomendas submetidas durante a sessão.	Medida calculada pela soma de todos os `Valor da Encomenda em Euros` existente ao nível da tabela para análise de utilização de páginas *Web.*	Soma.

Tabela 6.7 – Factos medidos da Tabela de factos para análise de sessões completas

A granulosidade dos dados ao nível dos *logs* dos servidores *Web* corresponde a um registo por pedido HTTP. Embora este nível de detalhe possa ser transposto para o *Data Webhouse*, eventualmente não será necessário dispor de tanto pormenor. Com o aumento do grão chegou-se a uma agregação ao nível da página e ao nível da sessão. Após as dimensões e tabelas de factos adequadas a esse grão terem sido processadas poderá existir um processo de filtragem de dados. Aqui poder-se-á, por exemplo, descartar tudo o que for registos de imagens ou outros ficheiros ornamentais ou auxiliares.

Capítulo 7

Integração de Dados e Operação do Data Webhouse

7.1 O Processo de Integração e Rotinas de Manutenção

O processo final de transformação deve consolidar o formato dos dados. As dimensões e tabelas de factos na ZCD deverão ser coladas em tabelas com a mesma estrutura das existentes no *Data Webhouse*. Desta forma, o processo de integração de dados ficará simplificado e permitirá um carregamento de dados mais optimizado ao não ter de executar interrogações demasiado complexas. Todavia, a não ser que estejamos a lidar com volumes de dados reduzidos, deverá ser utilizado um programa de carregamento em bloco em detrimento de interrogações do tipo `SELECT * FROM Tabela_na_ZCD INTO Tabela_no_Webhouse`. Este tipo de carregamento já foi apresentado anteriormente para o carregamento de ficheiros no SGBD da ZCD. Todavia, mesmo que o carregamento de dados da ZCD para o *Webhouse* seja dentro do mesmo SGBD, obter-se-á benefício na utilização do carregamento em bloco devido à sua maior rapidez de execução. Uma forma de o fazer é escrever uma interrogação do tipo `SELECT * from Tabela_na_ZCD` e redireccionar os resultados directamente para o programa de carregamento em bloco.

Para SGBD Oracle poderia ser usado algo do género:

```
sqlplus utilizador\password@base_de_Dados < interrogação.sql |
sqlload parametros.ctl utilizador password
```

O indicador de redireccionamento indicado, '|', é adequado para ambientes Unix. Em ambientes Microsoft-DOS/Windows este terá de ser substituído por '>'. O ficheiro de parâmetros do `sqlload`, `parametros.ctl`, terá de indicar que este deverá receber dados a partir de um fluxo redireccionado.

Em SGBD da Microsoft pode ser usado, a partir da versão 2000 do *SQL Server*, o comando de importação/exportação dos *Data Transformation Services* (DTS). Se necessário, será também possível, a partir de uma solução mais rebuscada e de uma forma semelhante à apresentada para o SGBD Oracle, a utilização do comando `isql`, para executar interrogações à base de dados, e do `bcp`, para a inserção em bloco na base de dados. Ambos os comandos, `isql` e `bcp`, são invocáveis a partir da linha de comando do sistema operativo.

Os principais SGBD actuais mantêm *logs* e segmentos de *rollback* extremamente úteis na recuperação de falhas em bases de dados transaccionais de suporte ao negócio. No entanto, estas mesmas funcionalidades podem ser causa de atrasos ou de falhas no carregamento dos dados no *Webhouse* devido a dimensionamentos insuficientes. A funcionalidade de *logging* do SGBM deverá ser desligada durante o carregamento das tabelas do *Data Webhouse*. Se ocorrer alguma falha no carregamento teremos sempre os dados na tabelas da ZCD e será possível reiniciar a integração.

Em situações onde seja necessário o re-carregamento completo de uma tabela do *Data Webhouse* é necessário apagar primeiro todos os seus registos. Essa operação deverá ser feita, sempre que possível, com a utilização de um comando que permita apagar todos os registos de uma só vez. No SGBD Oracle e *Microsoft SQL Server* isto é conseguido com o comando `truncate table`. Esta forma é mais eficiente do que o comando SQL `delete * from tabela` já que poderá ser usada sem ter a necessidade de criar registos nos segmentos de *rollback*.

Durante o processo de integração de novos dados poderá haver períodos onde se verifique uma violação temporária das regras programadas no SGBD. Estas deverão pois ser desactivadas no início do processo já que podem eventualmente bloquear todo o carregamento. Após a integração estar terminada deverão ser de novo activadas.

Durante o processo de integração nem todos os registos serão novos nas dimensões. Terá de ser implementado um processo que lide correctamente com as alterações a registos já existentes, seja pela rescrita, pela criação de novos registos ao nível das dimensões, ou pela adição de um novo registo onde os antigos valores são mantidos em atributos específicos.

Convém relembrar que será certamente necessária a criação de dois processos de integração de dados no *Data Webhouse*. Um para a primeira vez e outro para as seguintes. A primeira integração deverá preocupar-se em preparar todas as dimensões para conterem pelo menos um registo que possa ser usado para valores desconhecidos ou corruptos. Isto para poder manter a integridade referencial entre as tabelas de factos e as várias dimensões. Poderão ocorrer situações onde um processo terá falhado na obtenção de dados de uma dada fonte dentro da janela de oportunidade. Todo o processo de ETI poderá abortar ou então poderá estar preparado para utilizar um destes registos existentes ao nível das dimensões que permitam indicar que um determinado valor é desconhecido. Esta situação poderia ser então corrigida durante a próxima integração de dados.

Por forma a optimizar o acesso, permitindo uma maior rapidez na execução de interrogações ao *Webhouse*, devem ser criados alguns índices que se reconhece à partida terem um impacto positivo no processamento destas interrogações. Como ponto de partida para a indexação pode ser seguida uma regra simples: indexar individualmente todas as colunas com probabilidade de serem usadas como condições de junção, filtragem ou agrupamento. Este será o caso de quase todos os atributos que formam a chave primária das tabelas de factos. Relembremos também que os atributos declarados como chave primárias são automaticamente indexados pelo SGBD. Os índices são, contudo, uma "*faca de dois gumes*" em termos de desempenho. Quantas mais inserções e remoções existirem numa tabela mais recursos serão despendidos nas actualizações dos diversos índices que possam existir sobre os atributos da mesma. Se durante um processo de carga do *Data Webhouse* forem adicionados mais de 10 a 20 por cento ao tamanho de uma tabela então será mais eficiente, em termos de tempo, remover primeiro os índices existentes sobre a mesma e no

final voltar a recriá-los [Kimball et al. 98]. Caso se opte por manter os índices activos durante o processo de integração de dados no *Webhouse* então convém que estes sejam correctamente dimensionados para suportarem o volume de dados carregados. Isto para além das preocupações necessárias à distribuição dos mesmos pela infra-estrutura de *hardware* por forma a ficarem em discos diferentes daqueles onde se situam os dados que estão a ser indexados.

Deverá ser feita uma monitorização da utilização do *Data Webhouse* durante o processo normal de operação, nomeadamente, sobre o tipo de interrogações ao sistema que ocorrem com maior frequência, e sobre que tabelas e atributos, para decidir se será necessária a criação adicional de índices.

Para além dos índices, poderão ser notadas necessidades de criar agregações adicionais de dados, para optimizar as interrogações feitas sobre o *Data Webhouse*. Em [Joshi et al. 03] foi sugerida uma agregação dos dados em períodos temporais de um mês. Numa agregação deste género a, ou as, tabelas de factos poderão funcionar de duas maneiras:

- É criada uma nova agregação apenas quando o período de tempo para o qual se está a agregar dados termina.
- A tabela com a agregação do período mais recente vai sendo aumentada com os novos dados à medida que estes são integrados do género "mês até à data actual".

Um processo que poderá ainda ser feito após a integração de novos dados é a remoção e arquivo de dados que já não estejam dentro do período temporal de análise que se pretenda manter no *Data Webhouse*.

7.2 Exploração de *Data Webhouses* Através de Técnicas de Processamento Analítico

Depois dos dados estarem carregados na *Webhouse*, a integridade verificada e a qualidade assegurada, os utilizadores esperam poder ter à sua disposição ferramentas que permitam efectuar

análises *ad hoc* com tempos de resposta imediatos. É aqui que as técnicas de processamento analítico podem entrar.

O *Online Analytical Processing* (OLAP) surgiu exactamente com esse objectivo: permitir as interrogações *ad hoc* com elevado desempenho. O OLAP foi desenvolvido por forma a poder trabalhar eficientemente com dados organizados segundo o modelo dimensional encontrado nas *Data Warehouses*.

As *Data Warehouses* permitem uma visão multidimensional dos dados com um modelo intuitivo construído por forma a dar resposta às interrogações dos analistas. O OLAP reorganiza em cubos *multi-dimensionais* os dados presentes no *Data Warehouse* e depois processa esses cubos por forma a permitir o máximo desempenho nas várias interrogações sobre totais e sumários. Por exemplo, uma base de dados OLAP pode conter informações sobre vendas com agregações por produto, semana, região e canal de vendas. Após uma primeira análise da informação um utilizador poderia, tipicamente, efectuar operações de, entre outras, agregação (*drill-up*) ou de aprofundamento de análise (*drill-down*) numa das dimensões, efectuar selecções fatiadas criando mini-cubos em função de uma condição especificada (*slicing*), efectuar selecções numa dimensão baseadas em restrições sobre um atributo dessa dimensão.

Existem essencialmente três tipos de arquitecturas OLAP:

- OLAP Multi-dimensional (MOLAP ou MD-OLAP).
- OLAP Relacional (ROLAP), também chamado OLAP multi-relacional.
- OLAP Híbrido (HOLAP), também chamado *Managed Query Environment* (MQE).

As ferramentas MOLAP utilizam uma arquitectura que suporta o armazenamento dos dados em estruturas do tipo de matrizes *multi-dimensionais*, em vez de estruturas tabelares. Estas matrizes são carregadas a partir do SGDM relacional com agregações feitas em função da utilização prevista. Com ROLAP os dados são mantidos no SGBD relacional mas é criada uma camada de meta-dados que permite criar vistas multi-dimensionais das estruturas tabelares. Ao executar operações *multi-dimensionais* em sistemas ROLAP o que o sistema faz é traduzir essas operações em interrogações SQL que são executadas sobre o SGBD relacional. O HOLAP é uma mistura de ROLAP com MOLAP, permite tanto interrogações directas ao SGBD como a criação de cubos *multi-*

dimensionais que são transferidos para o sistema do utilizador. Uma descrição mais aprofundada sobre OLAP e os seus tipos de arquitectura pode ser consultada em [ConnolyBegg02] e [Franconi et al. 00]. O desempenho entre as várias arquitecturas OLAP perante algumas das funcionalidades requeridas por um *Webhouse* é variável (Tabela 7.1) e convém ser conhecido antes da arquitectura final ser escolhida [Sweiger et al. 02].

Funcionalidade	***MOLAP***	***HOLAP***	***ROLAP***
Tamanho da Clickstream Data Mart	Reduzida (apenas cubo de dados).	Elevado (cubos de dados e bases de dados relacionais).	Elevado (apenas bases de dados relacionais).
Desempenho de interrogação inicial	Elevado.	Moderado.	Imprevisível.
Tempo de resposta Drill/Pivot	Consistente.	Relativamente consistente.	Inconsistente (depende da interrogação).
Capacidade analítica avançada (análise de séries temporais, excepções estatísticas, etc.)	Elevada.	Elevada.	Baixa (limitado pela capacidade do SQL).
Eficiência no tempo de carregamento	Reduzida (pelo carregamento sequencial dos cubos).	Média (devido ao impacto dos cubos de dados).	Elevada (SQL paralelo e em bloco).

Tabela 7.1 – Comparação entre funcionalidades e tipos de OLAP

Genericamente, podemos efectuar dois tipos de análises sobre os registos de acessos ao sítio *Web*:

- Sumariar multidimensionalmente a informação proveniente dos *logs*.
- Derivar padrões e regras de utilização que suportem o negócio.

Eis alguns exemplos:

- **Análises de utilização:** O volume e distribuição dos acessos por tópicos específicos, dimensionados por computador de origem, referenciadores e tempo, podem ser usados como valores de referência por forma a poder efectuar uma personalização de conteúdos para os clientes de diferentes localizações e a horas diferentes.
- **Análise do tráfego do Servidor *Web***: dimensionados por servidor de origem e tempo, pode ser usado para planeamento de recursos e largura de banda bem como distribuição de cargas entre vários servidores, com recurso a réplicas e *caching* de conteúdos.
- **Descoberta de regras de negócio**: A evolução do número de acessos ao servidor *Web* pode fornecer indicações sobre a mudança de interesses e comportamento dos clientes. Por exemplo, a correlação entre um tópico e os acessos provenientes de uma certa origem descreve os interesses dos clientes dessa área. Enquanto que estes relacionamentos são úteis para ajudar a decidir sobre campanhas de *marketing*, as mudanças nestes relacionamentos podem ser ainda mais significativas já que tais mudanças podem reflectir em tempo real tendências nas evoluções dos interesses dos clientes, reacções a campanhas ou mesmo a influências da concorrência. Para capturarmos estes relacionamentos poderá ser necessário recorrermos a ajudas extra de ferramentas de *Data Mining* para a descoberta contínua de regras de associação.

Enquanto que arquitecturas da *Data Warehouse* com armazenamento OLAP são capazes de suportar volumes significativos de dados, o advento da *Web* e dos milhões de registos para análise geraram um desafio ainda maior. Com esta magnitude de dados, a computação dos cubos OLAP pode ser uma operação extremamente pesada e não conseguir ser feita dentro da janela de oportunidade. Em [Chen et al. 00a] é proposta uma arquitectura para lidar com o problema de escalabilidade nestas situações, nomeadamente no que diz respeito às operações de agregação em cubos *multi-dimensionais*.

Capítulo 8

Webuts – Web Usage Tracking Statistics

8.1 Contextualização e Âmbito

Para demonstrar a utilização de alguns dos elementos, técnicas, princípios e boas práticas estudadas e apresentadas ao longo desta dissertação foi desenhado, projectado e implementado, com recurso a um *Data Webhouse*, o *Webuts-Web Usage Tracking Statistics*. O *Webuts* é um protótipo de um sistema de apoio à decisão para acompanhamento e análise estatística das actividades dos utilizadores de um sítio *Web*.

O sítio *Web* alvo de estudo é o sítio corporativo da Sonae Indústria que pode ser acedido em http://www.sonaeindustria.com. A Sonae Indústria é a sub-holding do grupo Sonae que incorpora todos os negócios de produção e comercialização de produtos e componentes derivados de madeira. A diversidade de oferta comercial é grande e podemos citar alguns dos principais produtos: painéis de aglomerado de madeira, tanto em cru como revestidos, painéis de fibras de madeira, contraplacados de madeira, pavimentos, kits de mobiliário etc.

O sítio *Web* da Sonae Indústria é essencialmente um sítio de apresentação institucional. Este é destinado a prestar informação aos seus clientes, accionistas e público em geral. Para além da

informação de contacto e desempenho financeiro é apresentada toda a carteira de produtos que comercializa a nível mundial. É utilizado também para divulgar oportunidades de emprego dentro das diversas empresas da Sonae Indústria.

O sítio *Web* é multilíngue com versões das páginas em português, espanhol, inglês e francês. O público alvo encontra-se potencialmente distribuído pela Europa, América do Norte e do Sul, África e Médio Oriente.

Sendo apenas um protótipo, o *Webuts* não pretende incorporar todas as técnicas e boas práticas apresentadas e estudadas. Como o seu desenvolvimento é limitado pelo tempo e esforço disponível tal não seria possível. De igual forma, nenhuma das optimizações desejáveis no sítio *Web* por forma a simplificar o processamento de dados de *Clickstream* foi considerada. Contudo a construção deste protótipo pretende mostrar que, apesar destes constrangimentos, a organização detentora do sítio *Web* poderá obter informação previamente desconhecida que poderá ser utilizada para benefício próprio.

Pretende-se que através deste protótipo possam ser respondidas, entre outras, as seguintes questões:

- Sendo o sítio *Web* corporativo destinado a uma audiência global qual é a proveniência geográfica dos visitantes?
- Que partes do sítio *Web* são aquelas que mais visitas registam?
- Sendo a Sonae Indústria uma organização multinacional com milhares de funcionários, que proporção de visitas ao sítio *Web* é realmente gerado por utilizadores externos?
- Quais são os motores de pesquisa que mais tráfego fazem chegar ao sítio *Web*?
- Quais as palavras de pesquisa utilizadas por quem chegou ao sítio através de um motor de pesquisa?
- Que outros sítios *Web* estão a referenciar visitantes?
- Quantos visitantes são humanos e quantos são sistemas automáticos?
- Qual o tempo médio de duração de uma visita ao sítio *Web*?
- Qual a evolução do volume de tráfego ao longo do tempo?
- O volume de tráfego aumenta no sítio *Web* quando são publicados os relatórios de contas?
- Existem páginas *Web* que estão a dar origem a erros?

8.2 Fontes de Dados para o Sistema

Como já mencionado neste documento, ver capítulo 3, são muitas e diversas as fontes de dados de um *Data Webhouse*. Contudo, para o sítio *Web* em estudo, e não sendo este um sítio *Web* transaccional, foram escolhidas seis fontes de dados para permitir dar resposta às questões base colocadas pela organização, nomeadamente:

- Logs do servidor *Web* onde o sítio está alojado.
- Base de dados com informação da área geográfica à qual uma gama de endereços IP estão atribuídos.
- Lista de robots/agentes-automáticos conhecidos.
- Lista de motores de pesquisa.
- Lista de padrões de acessos perpetrados por vírus e tentativas de ataques a vulnerabilidades do servidor ou aplicações *Web*.
- Lista de computadores e redes internas à Sonae Indústria.

O servidor *Web* onde o sítio está alojado é um Microsoft IIS. Neste caso, os *logs* seguem o formato Microsoft IIS W3C *Extended* Log Format. Este sítio *Web* está alojado num servidor externo à organização, gerido por um prestador de serviços de alojamento. Eis um pequeno extracto do ficheiro de *log* gerado pelo servidor *Web*:

```
#Software: Microsoft Internet Information Services 5.0
#Version: 1.0
#Date: 2004-07-01 23:01:18
#Fields: date time c-ip cs-username s-sitename s-computername s-ip s-port
cs-method cs-uri-stem sc-status sc-bytes cs-bytes time-taken cs-host
cs(User-Agent) cs(Cookie) cs(Referer)
2004-07-01 23:01:18 200.152.225.91 - W3SVC13 WEBMASTER 195.23.88.11 80 GET
/banner/banner_brasil.htm   304   188   333   0   www.sonaeindustria.com
Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT+4.0)
ASPSESSIONIDACTSDARQ=NEGIEBCACCLBPFGJDNNOLDLA -
2004-07-01 23:01:18 200.152.225.91 - W3SVC13 WEBMASTER 195.23.88.11 80 GET
/banner/banner_brasil.gif   304   188   398   16   www.sonaeindustria.com
```

```
Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT+4.0)
ASPSESSIONIDACTSDARQ=NEGIEBCACCLBPFGJDNNOLDLA
http://www.sonaeindustria.com/banner/banner_brasil.htm
2004-07-01 23:01:18 200.152.225.91 - W3SVC13 WEBMASTER 195.23.88.11 80 GET
/banner/banner_ponto.gif    304    188    397    15    www.sonaeindustria.com
Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT+4.0)
ASPSESSIONIDACTSDARQ=NEGIEBCACCLBPFGJDNNOLDLA
http://www.sonaeindustria.com/banner/banner_brasil.htm
```

Tipicamente existem duas formas de determinar qual é a origem geográfica de um pedido HTTP que chega ao servidor *Web*:

- Através do domínio do computador que efectuou o pedido.
- Pelo detentor do endereço IP utilizado para fazer esse pedido.

A informação que se consegue obter a partir do domínio é condicionada pela forma como o emissor do pedido registou o seu domínio. Se este for do tipo xpto.pt então podemos inferir que o acesso é feito por um computador localizado em Portugal. Contudo, se o computador do utilizador for proveniente de uma rede que utilize um dos chamados domínios de topo, por exemplo **.com, .biz, .net, .org**, ou mesmo um dos domínios vendidos pelos seus legítimos detentores para utilização internacional, por exemplo **.tv** ou **.ws**, então não será possível determinar com exactidão, e de uma forma automatizada, qual o verdadeiro país, ou região, de onde o pedido HTTP provém. Uma condicionante adicional é a necessidade de efectuar a tradução dos endereços IP's registados nos *logs* do servidor *Web* no seu nome completo de domínio. Esta operação, embora possível, está quase sempre desactivada nos servidores *Web* devido ao acréscimo de processamento exigido. Ora, se esse processamento não foi feito a nascente então é necessário ser feito a jusante e nesse caso será o *Data Webhouse* que terá de suportar a carga da tradução destes endereços IPs nos seus nomes de domínio.

A outra forma de identificar a proveniência geográfica de um pedido é através da informação contida na base de dados `whois`. Esta base de dados é partilhada e publicada internacionalmente em vários servidores acessíveis pela Internet. Contudo, alguns destes servidores têm vindo a tornar o acesso a essa informação mais restrito através da limitação, por exemplo, do número de

interrogações que podem ser feitas à base de dados dentro de um dado período a partir do mesmo endereço IP. Esta é uma forma de combater a utilização abusiva da informação contida na base de dados para fins ilícitos. Como exemplo disso, temos o envio de mensagens de correio electrónico em massa para os endereços dos contactos administrativos e técnicos associados a um registo de um IP.

Existem, no entanto, diversos servidores que ainda se mantêm livres de restrições de acesso. Nessas situações poderia ser usado o comando `whois`, disponibilizado em quase todos os sistemas operativos, para obtenção de informação geográfica associada ao IP. Do uso do comando `whois` adviria também um acréscimo do tempo de processamento. A forma escolhida para uso no *Webuts* é uma base de dados mantida e disponibilizada gratuitamente pela empresa Maxmind [MaxGeoIP] no seu sítio *Web*. Esta base de dados é disponibilizada sob a forma de ficheiro e pode ser descarregada e acedida localmente. Desta forma, reduz-se o tempo necessário para obter a informação do país, ou região, associado a um endereço IP. Podemos ver no exemplo que se segue um extracto da informação contida neste ficheiro:

```
"2.6.190.56","2.6.190.63","33996344","33996351","GB","United
Kingdom"
"3.0.0.0","4.17.142.255","50331648","68259583","US","United States"
"4.17.143.0","4.17.143.15","68259584","68259599","CA","Canada"
"4.17.143.16","4.18.32.71","68259600","68296775","US","United
States"
"4.18.32.72","4.18.32.79","68296776","68296783","MX","Mexico"
```

O conteúdo traduz-se num limite inferior e outro superior de uma gama de endereços IP, os seus equivalentes decimais, e a região à qual estão atribuídos. O valor decimal de um endereço IP é calculado pela fórmula:

```
dec = (((((octeto1 * 256 + octeto2) * 256) + octeto3) * 256) +
octeto4)
```

É verdade que a fidelidade da informação fica dependente da velocidade com que esta base de dados é actualizada, mas o que se poupa em esforço e tempo de processamento justifica a sua utilização. Enquanto que com a utilização do comando `whois` se demoraria cerca de um ou dois segundos para obtenção da informação geográfica associada a um endereço IP, recorrendo ao uso desta base de dados, após ter sido carregada localmente, reduz-se esse tempo para apenas alguns mili-segundos.

A identificação dos robots/agentes-automáticos é feita com recurso à informação disponibilizada pela robotstxt.org. Esta organização mantém uma lista com um conjunto de diversas características associadas a cada robot/agente-automático. Estas características são fornecidas pelos seus criadores ou operadores e as mais importantes serão, neste protótipo, os endereços de proveniência, indicado pela etiqueta `robot-host`, e o texto identificador do robot que é enviado no cabeçalho HTTP, indicado pela etiqueta `robot-useragent`. Serão os valores destas características que ajudarão a identificar e classificar automaticamente os acessos ao servidor *Web* como tendo sindo efectuados por utilizadores humanos ou não humanos. Enquanto os endereços IPs, ou nomes dos computadores dos utilizadores, registados nos *logs* do servidor *Web* poderão ser comparados com o o valor contido na etiqueta `robot-host`, o valor contido na etiqueta `robot-useragent` poderá ser usado para comparação com o campo `user-agent` desses mesmos *logs*.
De seguida podemos encontrar um exemplo da informação disponibilizada sobre um robot:

```
robot-id:           ahoythehomepagefinder
robot-name:         Ahoy! The Homepage Finder
robot-cover-url:    http://www.cs.washington.edu/research/ahoy/
robot-details-url:
http://www.cs.washington.edu/research/ahoy/doc/home.html
robot-owner-name:   Marc Langheinrich
robot-owner-url:    http://www.cs.washington.edu/homes/marclang
robot-owner-email:  marclang@cs.washington.edu
robot-status:       active
robot-purpose:      maintenance
robot-type:         standalone
robot-platform:     UNIX
robot-availability: none
```

```
robot-exclusion:    yes
robot-exclusion-useragent: ahoy
robot-noindex:      no
robot-host:         cs.washington.edu
robot-from:         no
robot-useragent:    'Ahoy! The Homepage Finder'
robot-language:     Perl 5
robot-description:  Ahoy! is an ongoing research project at the
                    University   of   Washington   for   finding   personal
Homepages.
robot-history:      Research project at the University of Washington in
                    1995/1996
robot-environment:  research
modified-date:      Fri June 28 14:00:00 1996
modified-by:        Marc Langheinrich
```

A lista de padrões de acessos perpetrados por vírus e tentativas de ataques a vulnerabilidades do servidor *Web* visa sobretudo permitir separar os acessos legítimos dos acessos de cariz ilegítimo. A informação contida nos *logs* do servidor *Web* poderá ser separada e tratada de acordo com o seu propósito. Se num acesso, por exemplo, tiver sido efectuada uma tentativa de invocação do ficheiro `cmd.exe` isto é um indicador de uma tentativa de explorar uma vulnerabilidade que esteve associada a algumas das versões do servidor IIS da Microsoft. Esta lista de padrões é mantida manualmente pelo administrador do *Webuts* num formato directamente utilizável pelo `logparser` [MicrosoftIIS6RKT03]. Eis um extracto dessa lista:

```
cs-uri-stem LIKE '%cmd.exe%' OR
cs-uri-stem LIKE '%root.exe%' OR
cs-uri-stem LIKE '%default.ida%' OR
cs-uri-stem LIKE '%httpodbc.dll%' OR
cs-uri-stem LIKE '%/nsiislog.dll%' OR
cs-uri-stem LIKE '%/admin.dll%' OR
cs-uri-stem LIKE '%null.idq%'
```

A lista de computadores e redes internas à organização detentora do sítio *Web* servirão para distinguir qual a origem dos pedidos: se feitos a partir de computadores pertencentes à

organização ou a partir de computadores externos à organização. Esta lista é mantida manualmente pelo administrador do *Webuts* e toma a forma de uma tabela de expressões regulares de nomes e endereços IP pertencentes a empresas do grupo Sonae que podem ser usadas para classificar a proveniência do acesso registado no ficheiro de *log* do servidor *Web* como sendo interna ou externa.

A lista dos motores de pesquisa é uma tabela na base de dados onde se incluem, para além dos grandes motores de pesquisa internacionais, alguns de dimensão mais localizada ou mesmo dedicados a um país ou área geográfica (Tabela 8.1). Esta lista é mantida manualmente pelo administrador do *Webuts* e a versão inicial foi compilada com informação disponibilizada pela *Search Engine Watch* [SearchEng] e por informação incluída no analisador de *logs* AWStats [AWStats].

Chave	***Nome motor pesquisa***	***Expressão regular para detecção através do URI***	***Expressão regular para IP***	***Descrição motor de pesquisa***
1klik	1Klik	1klik\.dk		Motor de pesquisa Dinamarquês
Abacho	Abacho	suchen\.abacho\.de		Motor de pesquisa Alemão
Alexa	Alexa	alexa\.com		Importante motor de pesquisa internacional
allesklar	allesklar.de	allesklar\.de		Motor de pesquisa Alemão
alltheweb	AllTheWeb	alltheweb\.com		Importante motor de pesquisa internacional
metaspiner	Metaspinner	metaspinner	212\.227\.33\.241	Motor de pesquisa Alemão

Tabela 8.1 - Exemplo da informação mantida sobre os motores de pesquisa

8.3 O Modelo Dimensional do *Webuts*

O modelo dimensional do *Webuts* (Figura 8.1) é composto por uma tabela de factos, `Pedidos HTTP`, acompanhada por nove dimensões que ajudam a definir o contexto do facto: `Data`,

`Tempo`, `Agente HTTP`, `Estado HTTP`, `Computador do Utilizador`, `Método HTTP`, `Referenciador`, `URI` e `Utilizador`. A tabela de factos escolhida para este protótipo tem um grão igual ao registado pelos servidores *Web* nos ficheiros de *log*: um registo por cada pedido HTTP efectuado ao sítio *Web*.

As dimensões e tabela de factos são bastante idênticas às já descritas no capítulo 4 pelo que não serão detalhadas neste capítulo. Convém dizer que o critério utilizado para definir um utilizador foi, neste caso, uma combinação de vários campos existentes no *log* (Tabela 8.2).

Campo	***Descrição***
c-ip	Endereço IP do cliente
cs-username	Nome do utilizador
cs(user-agent)	O cliente HTTP utilizado no pedido. Normalmente identifica o navegador *Web*

Tabela 8.2 – Identificação de um utilizador

O rastreio de utilizadores será na maior parte das vezes feito para utilizadores anónimos já que o preenchimento do campo do nome do utilizador é efectuado apenas para a àrea destinada à administração do sítio *Web*. Para os casos onde apenas existam valores para o endereço IP e agente do utilizador é assumido que alguns utilizadores poderão erradamente ser considerados um único nas circunstâncias já discutidas anteriormente. Este será um ponto de eventual futura melhoria.

Será durante a geração da dimensão Utilizador que se irá também determinar se este é do tipo automático ou não. Para isso será utilizada a comparação da informação registada no *log* do servidor *Web* com a informação registada na lista de robots fornecida pela *robotstxt.org*. Para além das identificações pela comparação dos valores na lista, será considerado também que se um dado utilizador efectuar alguma vez um pedido ao URI do ficheiro `robots.txt` então este será considerado como sendo do tipo automático.

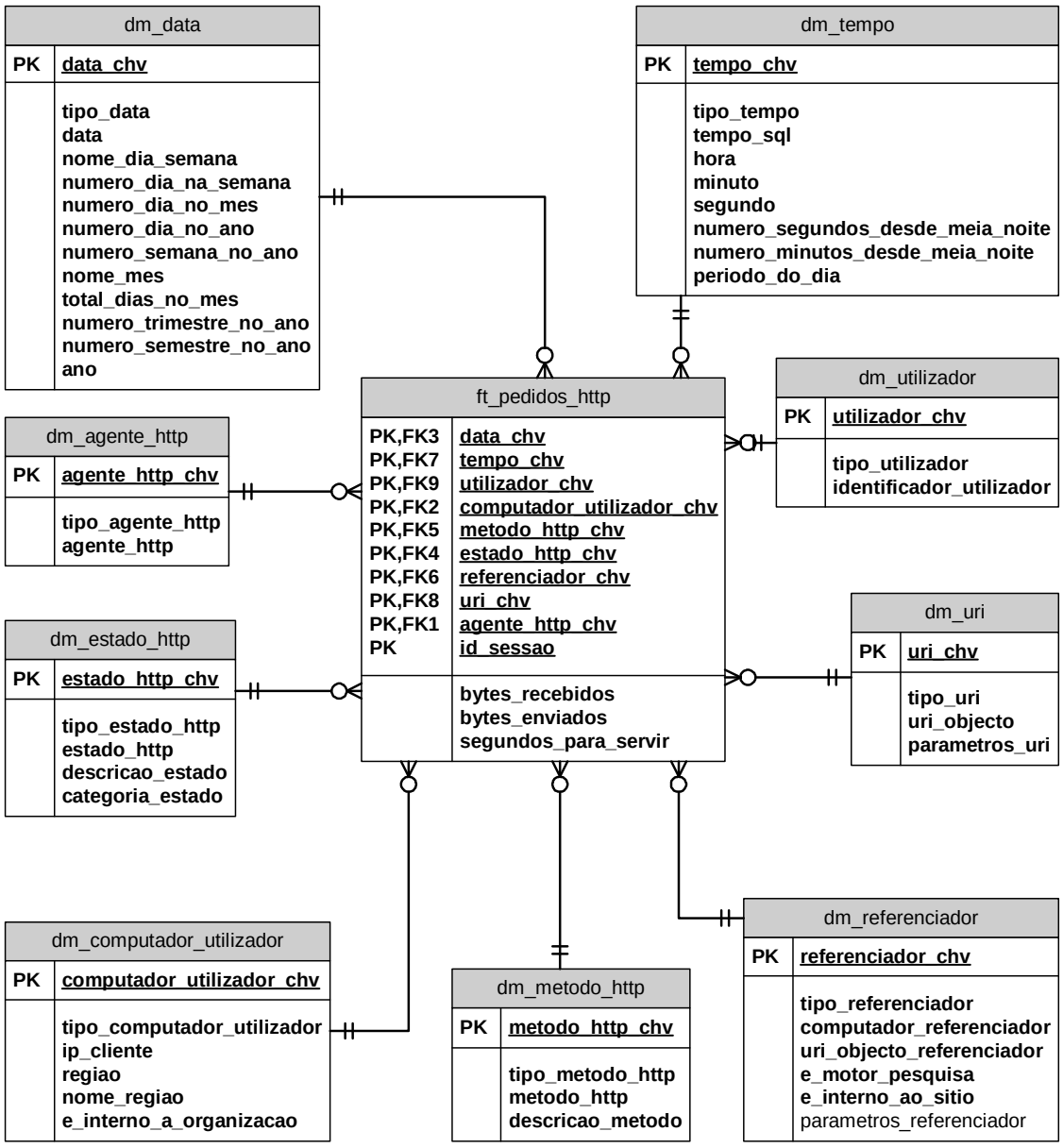

Figura 8.1 - Modelo dimensional do *Webuts*

Parte do sítio *Web* é construído com páginas html estáticas e outra parte com partes semi-dinâmicas onde é possível a alteração do conteúdo a partir da componente de administração do sítio *Web* mantendo, no entanto, o nome e estrutura da página. Estas páginas semi-dinâmicas são *Active Server Pages* (asp) e o conteúdo da página é armazenado numa base de dados relacional.

Neste tipo de aplicações asp o servidor *Web* da Microsoft faz a gestão automática de sessões registando o identificador de sessão numa *cookie* transiente. Contudo, esta é uma *cookie* que apenas será passada ao navegador do utilizador se este visitar uma dessas páginas asp. Se esta visita nunca acontecer e o utilizador apenas tiver navegado por páginas html estáticas então não haverá um registo da *cookie* de sessão nos ficheiros de *log*. Como tal, a identificação de sessão no Webhouse não poderá ser feita com recurso a esta *cookie*. A alternativa será a de considerar uma técnica reactiva para identificar a sessão, nos termos definidos em "Reconstrução de Sessões". Neste caso, a análise dos registos no *log* tem de determinar quais os pedidos efectuados pelo mesmo utilizador bem como identificar quando termina uma sessão e se inicia outra. O critério de término de sessão será temporal: se um utilizador não efectuar novo pedido ao servidor *Web* até 30 minutos após lhe ter sido servido o último pedido então considerar-se-á que futuros pedidos serão incluídos numa nova sessão. Este critério será auxiliado pela eventual existência, ou não, da *cookie* de sessão no registo do ficheiro de *log* que permitirá encurtar, ou alargar, a duração da sessão reconstruída pelo critério temporal.

8.4 O ETI do *Webuts*

Como já foi mencionado, o *Webuts* utiliza seis fontes distintas de dados externas e internas à organização (Tabela 8.3). A flexibilidade de escolha do mecanismo de colecta para o caso dos dados fora da organização é limitada. Em dois destes casos, fontes 2 e 3, o único mecanismo de transferência é baseado no protocolo HTTP.

As fontes de dados 4, 5 e 6 não existiam e foram criadas por forma a fornecer dados que permitam auxiliar na obtenção de respostas a algumas das questões colocadas pela organização. Estas fontes são mantidas manualmente pelo administrador do *Webuts* e são actualizadas sempre que necessário e justificável. A gestão e controlo das fontes de dados externas pela Sonae Indústria são diminutos no caso da fonte 1, e inexistentes no caso das fontes 2 e 3. Em face disto, o método de transferência terá de ser sempre do tipo *pull*, ou seja, a colecta será sempre despoletada a partir da ZCD. O ciclo de colecta e transferência de dados é baseado no período de rotação dos ficheiros de *log* do servidor *Web*, ou seja diário. Ainda no caso das fontes de dados 2 e 3, convém referir que não é possível uma passagem incremental de dados. Esta terá de ser

sempre completa já que não é possível detectar eficientemente na sua origem quais foram as alterações que essas fontes sofreram.

Tecnologicamente o *Webuts* foi implementado sobre uma base de dados *Microsoft SQL Server 2000* correndo sobre o sistema operativo Windows 2000 Professional. O *workflow* da componente de ETI do *Webuts* foi implementada com recurso aos *Data Transformation Services* (DTS) do *SQL Server*. Existem vários outros sistemas que poderiam ser utilizados para a implementação do *workflow* de ETI. A escolha recaiu, no entanto, sobre o *Microsoft SQL Server* pois este é o SGBD usado na Sonae Indústria para ambientes *Web*. A utilização desta tecnologia a nível de um protótipo iria facilitar o eventual desenvolvimento de um futuro sistema produtivo devido à experiência adquirida.

Este foi dividido em *workflows* mais pequenos, cada um associado a um processamento específico:

- ETI P1: Colecta e carregamento na ZCD dos ficheiros de *log* do servidor *Web*.
- ETI P2: Colecta e carregamento na ZCD do ficheiro com a informação geográfica associada aos endereços IP.
- ETI P3: Colecta e carregamento na ZCD do ficheiro com a informação da lista de robots conhecidos mantida pela robotstxt.org.
- ETI P4: Processamento das dimensões e geração da tabela de factos na ZCD.
- ETI P5: Integração no Webhouse das dimensões e tabela de factos.

O despoletar da execução do *workflow* de ETI é agendado através do calendário de *jobs* programável do *SQL Server*. A execução, neste caso, é calendarizada para logo após a rotação dos ficheiros de *log* do servidor *Web* em análise.

As diferentes tarefas do DTS são implementadas recorrendo ao suporte de diversas tecnologias:

- Programas diversos criados em T-SQL, ex: processamento das dimensões.
- Tarefas com função pré-definidas disponibilizadas pelo DTS, ex: *Bulk Insert* .
- Execução de ficheiros *batch* para invocação de programas ou comandos do sistema operativo, ex: colecta remota de ficheiros, descompressão.
- Extensões aos procedimentos *standard* do *SQL Server* com inclusão de funcionalidades externas, ex: permitir expressões regulares dentro do *SQL Server*.

No	***Fonte de Dados***	***Localização***	***Formato***	***Mecanismo de transferência***	***Tipo de passa-gem***
1	***Logs do servidor Web onde o sítio está alojado***	Externo à organização.	Ficheiro ASCII, *log* IIS W3C.	HTTP ou ftp.	Incremen-tal.
2	***Base de dados com informação da área geográfica dos IPs***	Externo à organização.	Ficheiro ASCII, tipo CSV, comprimido	HTTP.	Completa.
3	***Lista de robots/agentes-automáticos conhecidos***	Externo à organização.	Ficheiro ASCII etiquetado, terminadores de linha formato *Unix*.	HTTP.	Completa.
4	***Lista de motores de pesquisa***	Interno à organização.	Tabela em base de dados *SQL Server*.	T-SQL	Incremen-tal.
5	***Lista de padrões de acessos perpetrados por vírus e tentativas de ataques a vulnerabilida-des do servidor Web***	Interno à organização.	Ficheiro ASCII directamente utilizável pelo `logparser`.	Vários.	Incremen-tal.
6	***Lista de computadores e redes internas à organização detentora do sítio Web***	Interno à organização.	Tabela em base de dados *SQL Server*.	T-SQL.	Incremen-tal.

Tabela 8.3 – Métodos e mecanismos de transferência de dados das fontes do *Webuts*

Neste protótipo procurou-se minimizar os desenvolvimentos recorrendo, sempre que possível, e com excepção do SGBD, a programas e utilitários disponíveis gratuitamente na Internet. Para a execução de interrogações foi considerado para este protótipo a utilização do *Microsoft SQL Query Analyser*.

O ETI P1 (Figura 8.2) começa por colectar os ficheiros de *log* do servidor *Web*. Esta colecta é feita com recurso ao utilitário `wget` [UnixUtil]. Este pequeno utilitário, disponível em sistemas

operativos Unix e também em versões para o ambiente MS Windows, pode colectar remotamente objectos, usando o protocolo ftp ou o HTTP, bastando indicar qual, ou quais, os URIs desses mesmos objectos. O carregamento na base de dados da ZCD dos ficheiros de *log* colectados é feito recorrendo a outro utilitário gratuíto, o `logparser` [MicrosoftIIS6RKT03]. Este programa permite interrogações do tipo SQL sobre os ficheiro de *log* e escrita em bloco directamente numa tabela do *SQL Server*. Permite também a execução de um conjunto de funcionalidades interessantes como, entre outras, a tradução de endereços IP em nomes e a descodificação de caracteres do URI. Os acessos registados nos *logs* do servidor *Web* que se consideram poderem ter sido feitos com fins ilícitos ou dolosos, tipo ataques feitos por vírus, são neste carregamento separados e colocados numa tabela dentro da ZCD para posterior inspecção pelo administrador do *Webuts*. Após terem sido carregados para a base de dados da ZCD, os *logs* são normalizados e colocados numa tabela de formato neutro. Isto para que se lhe possam adicionar outros *logs* com outros formatos que entretanto poderão também surgir.

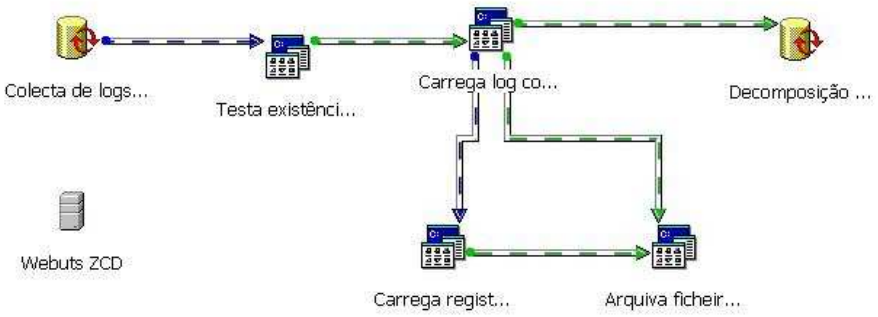

Figura 8.2 - ETI P1: Colecta e carregamento na ZCD dos ficheiros de *log*

A colecta remota do ficheiro com a informação geográfica associada aos endereços IP (Figura 8.3) é também efectuada com recurso ao wget. Neste caso, como é necessário uma passagem completa do ficheiro, é usada uma optimização na transferência: se a etiqueta temporalda última modificação do ficheiro remoto for igual à etiqueta temporaldo último ficheiro colectado então não é feita uma nova colecta. Neste caso, o ficheiro recolhido anteriormente é reutilizado mas uma optimização deste protótipo poderia passar por terminar o processo nesse ponto e não executar os passos seguintes do *workflow*. Após ser colectado, o ficheiro é descomprimido com o `unzip`

[UnixUtil], mais um utilitário disponível gratuitamente. De seguida são retiradas as aspas, que delimitam os valores das colunas dentro do ficheiro, recorrendo ao `sed` [UnixUtil]. Após essa limpeza, os dados são colocados na ZCD através de uma inserção em bloco. Estes novos dados irão substituir completamente os que já se encontravam na base de dados da ZCD.

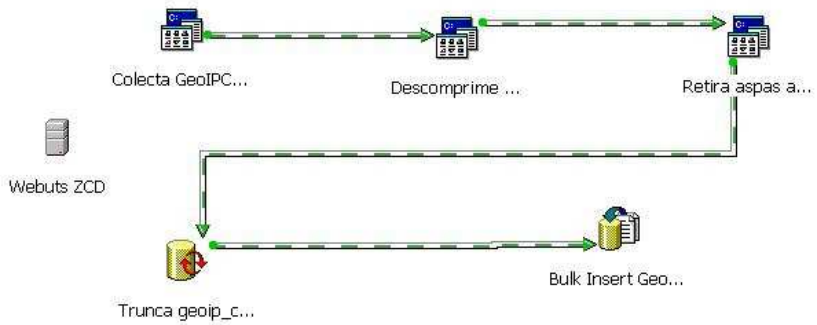

Figura 8.3 – ETI P2: Colecta e carregamento do GeoIP na ZCD

A colecta da informação sobre robots (Figura 8.4) é feita, mais uma vez, com recurso ao `wget`. Esta informação está num ficheiro de texto que tem a particularidade de ter terminadores de linha com formato Unix. Estes terminadores de linha são, então, convertidos para formato Dos/Windows recorrendo ao utilitário `unix2dos` [Unix2Dos]. Após essa conversão, o ficheiro é inserido na base de dados da ZCD e processado por forma a separar em colunas a informação que tem etiquetada por linhas.

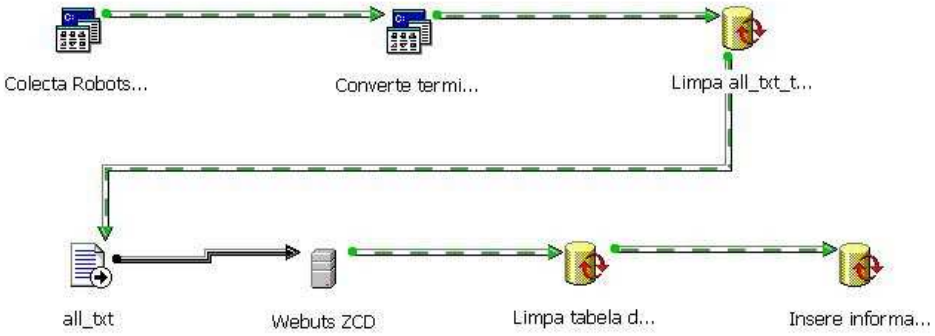

Figura 8.4 – ETI P3: Colecta e carregamento da informação sobre robots na ZCD

As dimensões de cariz estático, dimensão `Tempo`, `Método HTTP` e `Estado HTTP`, são geradas e carregadas manualmente na Webhouse pelo administrador logo após a sua geração. O processamento das dimensões dinâmicas é feito dentro do *workflow* ETI P4 (Figura 8.5). Todas as dimensões incluem sempre um registo para ser usado no caso de valores desconhecidos. Este registo é gerado e inserido logo na primeira vez que a dimensão é usada. O processamento das dimensões pode ser feito em paralelo já que, para o modelo dimensional usado no *Webuts*, não existem quaisquer tipos de dependências entre elas. A Identificação de sessões terá de ser precedida pela identificação dos utilizadores já que, como mencionado, as sessões serão reconstruídas pelo agrupamento no tempo dos pedidos HTTP efectuados ao servidor *Web* pelo mesmo utilizador. Após o processamento de todas as dimensões, é finalmente gerada a tabela de factos `Pedidos HTTP`. Todas as tabelas de dimensões e de factos na ZCD têm uma estrutura igual às existentes no *Data Webhouse* (DW) por forma a facilitar e optimizar a cópia dos dados.

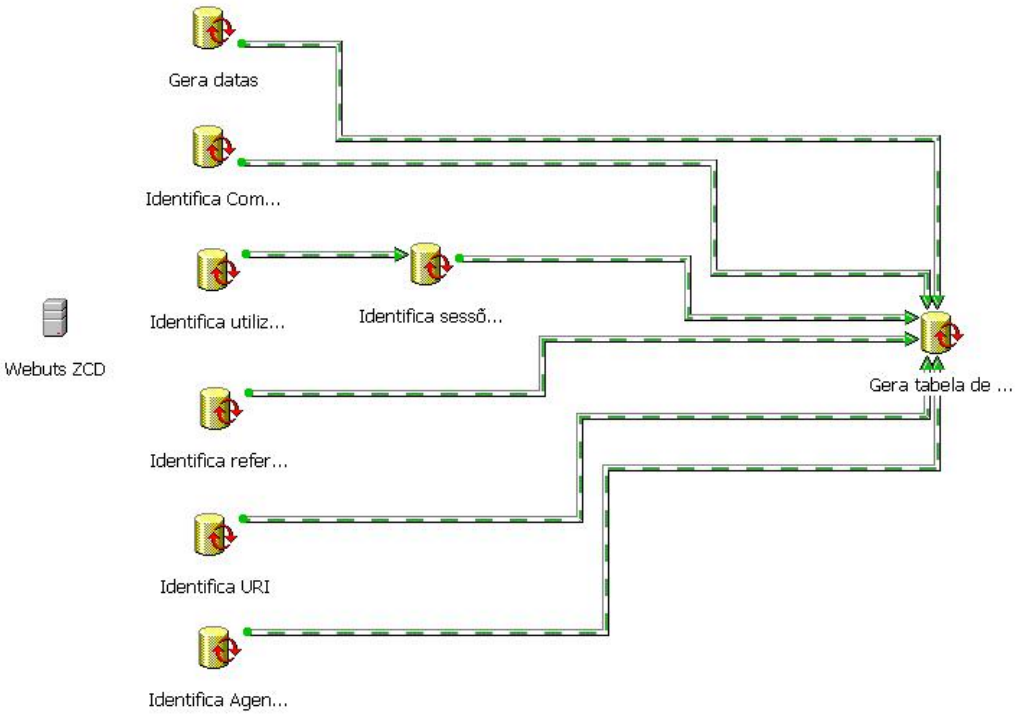

Figura 8.5 – ETI P4: Processamento das dimensões e geração da tabela de factos na ZCD

O *workflow* ETI P5 (Figura 8.6), efectua a integração dos dados no DW. Durante o processo de integração de novos dados poderá haver períodos onde se verifique uma violação temporária das regras programadas no SGBD. Estas regras são usados para validação e garantir a integridade dos dados. Neste caso, as regras de validação de chaves estrangeiras na tabela de factos são removidos no início do processo já que podiam eventualmente bloquear todo o carregamento. Após a integração estar terminada, estas são de novo recriadas. A passagem das dimensões e tabela de factos da base de dados ZCD para a base de dados do DW é feita em paralelo com recurso à *Transform Data Task* do DTS usando a simples função de cópia de registos. Neste protótipo não foram consideradas alterações a registos já existentes no DW. Todos os registos transferidos da ZCD para o DW são inserções de novos dados.

Após todos os dados terem sido integrados no DW com sucesso, é efectuada uma limpeza de todas as tabelas da ZCD usadas temporariamente durante o processo de ETI. Informações sobre essas tabelas e de todas as outras utilizadas no processo podem ser consultadas no Anexo III.

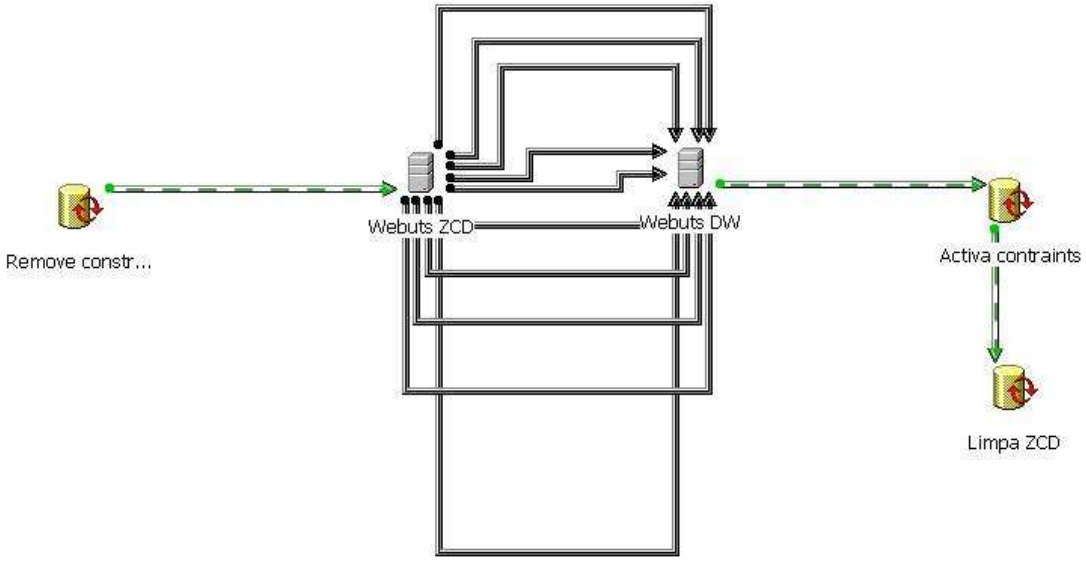

Figura 8.6 – ETI P5: Integração no Webhouse das dimensões e tabela de factos

Capítulo 9

Conclusões e Trabalho Futuro

Para uma organização, a motivação da implementação e utilização de um *Data Webhouse* passa pela vontade em conhecer de uma forma fidedigna quem são e o que pretendem os utilizadores do seu sítio *Web*. Isto por forma a poder satisfazer de uma forma rápida e cativante os interesses do visitante e daí tirar proveito para a organização, seja este de cariz financeiro ou outro. No entanto, passar de uma situação onde temos um registo, porventura incompleto, de quais os objectos e páginas *Web* pedidas pelo utilizador para uma situação onde possamos afirmar que o senhor X ou o senhor Y gostaram de visitar um dado sítio *Web* é, no mínimo, difícil de alcançar. Isto acontece, em grande parte, devido às limitações da própria tecnologia envolvida na operação dos sítios *Web*.

O sítio *Web* passou a integrar informação e funcionalidades fornecidas por distintos sistemas e entidades tanto internos como externos à organização. São inúmeras as aplicações, tecnologias e componentes que possibilitam ao utilizador o contacto com o sítio *Web* pela simples utilização de um navegador *Web*. Será deste conglomerado de elementos, com funcionamento mais ou menos interligado mas nem sempre standardizado, conjuntamente com os sistemas operacionais mais tradicionais, onde se irá buscar os dados necessários para se poder ter a visão global e integrada das actividades de um utilizador nos diversos pontos de contacto com a organização, com particularidade nas suas visitas ao sítio *Web*. Foram vários os sistemas descritos que poderiam funcionar como fontes de dados: servidores *Web*, *proxies*, *firewalls*, servidores multimédia e aplicacionais, motores de pesquisa, navegador e computador do visitante, *logs* de ISPs, dados de

servidores de redes publicitárias, sistemas transaccionais de suporte ao negócio, sistemas de gestão de contacto e, finalmente, dados demográficos e de mercado. Será no seio de um *Data Webhouse*, ponto de convergência e integração das diversas fontes de dados, onde melhor se farão as análise pretendidas pelos utilizadores finais.

Foi neste sentido que se apresentou um modelo dimensional para um *Data Webhouse* vocacionado para as análises num ambiente *Web* transaccional, embora muitas variações sejam possíveis. Foram apresentadas três tabelas de factos com diferentes objectivos e diferentes grãos:

- **Análise de Pedidos HTTP**: Esta tabela tem o mesmo grão que se encontra directamente registado nos ficheiros de *log* dos servidores *Web*. Este grão é o ideal para análises de cariz mais técnico necessárias para a administração dos sítios *Web*.
- **Análise de Utilização de Páginas *Web***: Esta tabela representa já uma agregação relativamente ao que se encontra nos *logs* do servidor *Web*. A utilização das páginas é, nesta tabela, directamente relacionada com as actividades que se podem registar em relação ao cesto de compras.
- **Análise de Sessões Completas**: Esta tabela é uma agregação da anterior e funciona como um sumário de cada visita dos utilizadores. Está vocacionada para o estudo a mais alto nível da utilização do sítio *Web*.

Estas tabelas de factos poderão ser utilizadas em conjunto ou separadamente. A escolha deverá ser feita em função das necessidades específicas de análise dos utilizadores finais.

Os dados de *clickstream* necessários para a construção do *Data Webhouse* são provenientes de diversas fontes. Estes dados são, na sua maioria, registados em ficheiros de texto sob a forma de *logs* de actividade mas com diferentes formatos. Esta heterogeneidade das fontes de dados obriga a operações de mapeamento, transformação e integração por forma a poder criar um formato homogéneo. Em *logs* de actividade é preciso ter uma noção clara do momento e sequência no tempo em que essas actividades ocorreram. Devido a isso, é necessária a utilização de um *standard* temporal seguido por todas as fontes geradoras de dados e com um valor perfeitamente sincronizado entre todas elas. Como se viu em exemplo apresentado, uma falta de sincronização temporal pode deturpar completamente, ou mesmo tornar inconclusiva, uma análise de comportamento num sítio *Web* suportado por múltiplos servidores *Web*.

O processo de extracção, transformação e integração dos dados de *clickstream* é cheio de desafios. Com dados gerados dentro e fora da organização, as decisões sobre os mecanismos e técnicas de colecta são variadas. Tudo depende do grau de controlo que existe sobre o sistema onde se situa a fonte de dados e do próprio tipo de dados. A periodicidade de recolha pode ser variada mas a integração no *Data Webhouse* é condicionada pela disponibilidade dos dados e tipicamente será em função do período de rotação dos *logs* dos servidores *Web*.

Antes de preparar as dimensões e tabelas de factos para integração é necessário, contudo, um conjunto de várias tarefas de pré-processamento dos dados de *clickstream*:

- Identificação de Utilizadores e Agentes Automáticos.
- Identificação de Sessões.
- Identificação de Páginas.

Estas são tarefas que podem ter, ou não, a execução facilitada em função da própria estrutura e funcionamento do sítio *Web*. A utilização de *cookies* e autenticação de utilizadores pelo sítio *Web*, por exemplo, são elementos que facilitarão imenso estas tarefas. O uso de *frames* e de páginas geradas dinamicamente, por outro lado, dificultarão a execução destas tarefas. Há situações em que dados terão de ser inferidos. Isto verificar-se-á na ausência de elementos que permitam dizer definitivamente se foi um utilizador A ou utilizador B quem efectuou um pedido, ou se um utilizador fechou o navegador e terminou a sessão ou simplesmente foi atender o telefone, ou ainda quando for necessário identificar qual, ou quais, as páginas que foram servidas a um utilizador a partir de uma qualquer *cache*. Contudo, há que ter a noção que os resultados finais irão ser directamente condicionados pela técnica de pré-processamento utilizada.

O processo de integração em si tem de estar optimizado para poder lidar com o potencialmente imenso volume de dados. As técnicas de integração de dados em bloco são recomendáveis e as funcionalidades comummente disponibilizadas pelos SGBD de *rollback* e *logging* deverão ser desactivadas durante esta operação. A indexação deve ser usada como forma de optimização de interrogações ao *Data Webhouse*. No entanto, os índices deverão ser apagados antes da integração de dados e recriados no final quando o volume de novos dados for elevado.

O protótipo *Webuts* foi apresentado como exemplo de um Webhouse para análise de um sítio *Web* não trasaccional e com o objectivo de dar respostas específicas sobre o sítio *Web* objecto de estudo. De âmbito limitado, o desenvolvimento do *Webuts* não utilizou todas as técnicas e boas práticas descritas ao longo de toda a dissertação. Foi mesmo assumida a utilização de técnicas que poderão gerar resultados incompletos. De igual forma, não foi feito nenhum melhoramento ao sítio *Web* que permitisse a obtenção de dados mais completos e fidedignos. Contudo, a construção deste protótipo possibilitou, apesar destes constrangimentos, mostrar à organização detentora do sítio *Web* que, através da utilização de um *Data Webhouse*, pode obter informação previamente desconhecida sobre o seu sítio que poderá utilizar para benefício próprio.

Várias melhorias são possíveis no *Webuts*. As dimensões são simples e poderão ser enriquecidas com mais atributos que permitam um estudo mais aprofundado de cada visita. O grão da tabela de factos é igual ao registado nos *logs* do servidor *Web*: um registo por cada pedido HTTP. Embora útil de um ponto de vista técnico este tipo de grão não é o adquado para análises de cariz mais sumárias. Os dois outros grãos apresentados nesta dissertação - um registo por cada página *Web* e um registo por cada sessão completa - dariam para efectuar mais análises do tipo navegacional ou comparativas entre visitas de uma forma mais eficiente .

O *Webuts* também não apresenta soluções no que diz respeito à recuperação de erros na execução dos *workflows* de ETI limitando-se a registar esse erro no ficheiro `webuts.log`. Na maioria das situações, se ocorre um erro o *workflow* simplesmente aborta a sua execução. Esta situação deverá ser também um ponto a melhorar no *Webuts*. Outro ponto onde o *Webuts* pode ser melhorado é no tratamento de diferentes formatos de *log* de servidores *Web*. Os desenvolvimentos centraram-se apenas no tratamento do formato *Microsoft IIS W3C Extended Log Format*.

A identificação e classificação de utilizadores é um ponto onde maior correcção poderá se incutida com a utilização de técnicas complementares para além das utilizadas. Estas técnicas, descritas no capítulo 6, ajudarão a aumentar o grau de certeza na distinção entre utilizadores. Ao aumentar a capacidade de distinção entre utilizadores, também aumentará o grau de fidelidade na reconstrução das sessões dos mesmos.

Nesta dissertação, também não foram mencionados, todavia, de uma forma concreta e abrangente os aspectos relacionados com a autenticação e autorização de acessos no *Data Webhouse*. Sendo o *Data Webhouse* o repositório da "verdade" de uma organização, a segurança é um dos componentes onde não se pode compactuar com amadorismos e que terá certamente o seu lugar em qualquer plano de implementação. Se se verificasse alguma falha de segurança com um acesso indevido por algum elemento interno à organização, esta poderia ser séria mas de consequências limitadas. Contudo, numa situação onde se poderá abrir o *Data Webhouse* a parceiros comerciais externos, qualquer falha na segurança poderá assumir proporções catastróficas tanto em termos de imagem como financeiramente. Eventualmente, poderá mesmo pôr em risco a sobrevivência da organização.

A construção de um *Data Webhouse* é uma área complexa e sem dúvida um desafio. Como se pode constatar ao longo de todo o documento, para além dos problemas comuns aos projectos dos *Data Warehouses* mais tradicionais, há uma panóplia de problemas que podem surgir num projecto deste tipo e quase ofuscar todos os outros: o volume de dados, a incompletude dos dados, a identificação unívoca dos visitantes de um servidor *Web*, etc.

Nesta dissertação de mestrado foram abordadas áreas e temas de grande importância para quem implementa, ou pensa em implementar, um *Data Webhouse*. Entendo que os assuntos abordados e o trabalho desenvolvido abrem caminho para novos projectos. Um desdes projectos será, após validação a nível profissional dos resultados obtidos, o desenvolvimento a curto prazo de um sistema de apoio à decisão, com recurso a um *Data Wenhouse*, para análise integrada de todos os sítios *Web* da Sonae Indústria. Este sistema seria uma evolução do *Webuts* onde, para além das melhorias já descritas, teriam de ser também tidas em conta questões de escalabilidade e desempenho. A continuação da pesquisa na área e a publicação de novos artigos sobre a temática de *clickstream* e *Data Webhouses* será outro objectivo futuro. Se as condições se proporcionarem gostaria, certamente, de desenvolver o tema focado neste documento para servir de base a um eventual doutoramento na área.

Referências

[ApacheHTTP] "*Apache* HTTP Server Version 2.0 Documentation", *Apache* Software Foundation, http://httpd.apache.org/docs-project/

[AWStats] AWStats - Advanced *Web* Statistics, http://awstats.sourceforge.net

[Baglioni et al. 03] M. Baglioni, U. Ferrara, A. Romei, S. Ruggieri, F. Turini: "Preprocessing and Mining *Web Log* Data for *Web* Personalization". *Lecture Notes in Computer Science: 8ª Conferência Italiana em Inteligência Artificial*, Volume 2829, páginas 237-249, Setembro, 2003, http://citeseer.nj.nec.com/576555.html

[BernersLee94] T. Berners-Lee: "Universal Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide *Web*". *RFC1630*, Junho, 1994, http:// www.rfc-editor.org /rfc/rfc1630.txt

[BernersLee et al. 94] T. Berners-Lee, L. Masinter, M. McCahill: "Uniform Resource Locators (URL)". *RFC 1738*, Dezembro, 1994, http:// www.rfc-editor.org /rfc/rfc1738.txt

[BernersLee et al. 96] T. Berners-Lee, R. Fielding, H. Frystyk: "Hypertext Transfer Protocol - HTTP/1.0". *RFC 1945*, Maio, 1996, http:// www.rfc-editor.org /rfc/rfc1945.txt

[BernersLee et al. 98] T. Berners-Lee, R. Fielding, L. Masinter: "Uniform Resource Identifiers (URI): Generic Syntax". *RFC 2396*, Agosto, 1998, http:// www.rfc-editor.org /rfc/rfc2396.txt

[Berendt et al. 02] Bettina Berendt, Bamshad Mobasher, Miki Nakagawa, Myra Spiliopoulou: "The Impact of Site Structure and User Environment on Session Reconstruction in *Web* Usage Analysis". Em *Proceedings of Fourth International Workshop on Knowledge Discovery in the Web WEBKDD'2002 - Web Mining for Usage Patterns & Profiles*, Agosto, 2002

[Bonchi et al. 01] Francesco Bonchi, Fosca Giannotti, C. Gozzi, Giuseppe Manco, Mirco Nanni, D. Pedreschi, Chiara Renso e Salvatore Ruggieri: "*Web log* data warehousing and mining for intelligent *Web* caching". *Data & Knowledge Engineering*, Elsevier, Volume 32, Número 2, páginas 165-189, Novembro, 2001, http://citeseer.nj.nec.com/bonchi01*Web*.html

[Borges et al. 03] Eurico Borges, Anália Lourenço, Orlando Belo: "Evaluating the Impact of Information Sources Heterogeneity Inside *Data Webhousing* Systems". Em *Proceedings of the 7th WSEAS International Multiconference on CIRCUITS, SYSTEMS, COMMUNICATIONS and COMPUTERS*, Corfu, Grecia, Julho 7-10, 2003

[Bouzeghoub et al. 99] Mokrane Bouzeghoub, Françoise Fabret, Maja Matulovic-Broqué: "Modeling *Data Warehouse* Refreshment Process as a *Workflow* Application". Em *Proceedings of International Workshop on Design and Management of Data Warehouses* (DMDW'99), Heidelberg, Alemanha, Junho, 1999

[Buchner et al. 99] Alex G. Büchner, S.S. Anand, Maurice D. Mulvenna, J.G. Hughes: "Discovering Internet *Marketing* Intelligence through *Web Log* Mining". Em *Proceedings of the Unicom99 Data Mining & Data Warehousing: Realising the full Value of Business Data*, páginas 127-138, 1999

[Cardellini et al. 02] Valeria Cardellini, Emiliano Casalicchio, Michele Colajanni, Philip S. Yu: "The State of the Art in Locally Distributed *Web*-server". *ACM Computing Surveys*, volume 34, número 2, páginas 263-311, 2002

[CatledgePitkow95] Lara D. Catledge, James E. Pitkow: "Characterizing browsing behaviors on the World Wide *Web*". *Computer Networks and ISDN Systems*, Elsevier, volume 27, número 6, páginas 1065-1073, Abril, 1995.

[ChauChen03] Michael Chau, Hsinchun Chen: "Personalized and Focused *Web* Spiders". *Web Intelligence*, Springer-Verlag, páginas 197-217, Fevereiro, 2003

[Chen et al. 00a] Qiming Chen, Umeshwar Dayal, Meichun Hsu: "An OLAP-based Scalable *Web* Access Analysis Engine". *Data Warehousing and Knowledge Discovery: Second International Conference, Dawak 2000 London, UK*, Springer-Verlag, páginas 210-223, Setembro, 2000, http://citeseer.ist.psu.edu/chen00olapbased.html

[Chen et al. 00] Ye-Sho Chen, Bob Justis, Edward Watson: "*Web*-enabled *Data Warehouse*". *Handbook of Electronic Commerce*, Springer-Verlag, páginas 501-520, 2000

[Clickstream01] "Technical White Paper". *Clickstream* Technologies PLC, 2001, http://www.*clickstream*.com

[ConnolyBegg02] Thomas Connoly, Carolyn Begg: "Database Systems - A Practical Approach to Design, Implementation and Management". 3ª Edição , Addison-Wesley, 2002

[Cooley et al. 99] Robert Cooley, Bamshad Mobasher, Jaideep Srivastava: "Data Preparation for Mining World Wide *Web* Browsing Patterns". *Journal of Knowledge and Information Systems*, Springer-Verlag, número 1, páginas 5-32, 1999

[EmpowerGeog] "Age Rank Report". Empower Geographics -Demographic Analyzer, SRC, http://www.demographicsnow.com/AllocateOnline.dll?ShowPage=static/samples.htm

[Fielding et al. 99] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee: "Hypertext Transfer Protocol -- HTTP/1.1". *RFC 2616*, Junho, 1999, http://www.rfc-editor.org /rfc/rfc2616.txt

[FieldingIrvine95] R. Fielding, U.C. Irvine: "Relative Uniform Resource Locators". *RFC 1808* , Junho, 1995, http:// www.rfc-editor.org /rfc/rfc1808.txt

[Franconi et al. 00] Enrico Franconi, Franz Baader, Ulrike Sattler, Panos Vassiliadis: "Fundamentals of *Data Warehouses*, Capítulo 5- Multidimensional Data Models and Aggregations". Springer, páginas 87-107, 2000

[HallamBehlendorf96a] Phillip M. Hallam-Baker, Brian Behlendorf: "Extended *Log* File Format". *W3C Working Draft WD-logfile-960323*, 1996, http://www.w3.org/pub/WWW/TR/WD-logfile.html

[HallamBehlendorf96b] Phillip M. Hallam-Baker, Brian Behlendorf: "Extended *Log* File Format". *World Wide Web Journal: The Web Five Years After*, Volume 1, Número 3, O'Reilly & Associates, Setembro, 1996, http://www.w3journal.com/3/s2.hallam.html

[HuCercone04] X. Hu, N. Cercone: "A *Data Warehouse*/OLAP Framework for *Web* Usage Mining and Business Intelligence Reporting". *International Journal of Intelligence Systems and Information Processing*, Volume 19, Número 7, páginas 567-584, Julho, 2004, http://www.cis.drexel.edu/faculty/thu/My%20Publication/Journal-papers/JofIS/jis_tonynick.pdf

[Joshi et al. 99] Karuna P Joshi, Anupam Joshi, Yelena Yesha, Raghu Krishnapuram: "Warehousing and Mining *Web* Logs". Em Procedings of the Second International Workshop on *Web* information and Data Management, Novembro, 1999

[Joshi et al. 00] Anupam Joshi, Karuna Joshi, Raghu Krishnapuram: "On mining *Web* access *logs*". Em *Proceedings SIGMOD 2000 Workshop on Research Issues in Data Mining and Knowledge Discovery*, Dallas, páginas 63-69, 2000

[Joshi et al. 03] Karuna P. Joshi, Anupam Joshi, Yelena Yesha: "On Using a *Warehouse* to Analyze *Web* Logs". *Journal of Distributed and Parallel Databases*, Kluwer Academic Publishers, Volume 13, Número 2, páginas 161-180, Março, 2003

[Kimball et al. 98] Ralph Kimball, Laura Reeves, Margy Ross, Warren Thornthwaite: "The *Data Warehouse* Lifecycle Toolkit – Expert Methods for Designing, Developing and Deploying *Data Warehouses*". John Wiley & Sons Inc., 1998

[KimballMerz00] Ralph Kimball, Richard Merz: "The *Data Webhouse* Toolkit – Building the *Web*-Enabled *Data Warehouse*". John Wiley & Sons Inc., 2000

[KohaviParekh03] Ron Kohavi, Rajesh Parekh: "Ten Supplementary Analyses to Improve E-commerce *Web Sítios*". Em *Proceedings of Fifth International Workshop on Knowledge Discovery in the Web WEBKDD'2003 - Webmining as a Premise to Effective and Intelligent Web Applications*, páginas 29-36, Agosto, 2003

[Koster94] Martin Koster: "A Standard for Robot Exclusion". 1994, http://www.robotstxt.org/wc/norobots.html

[KrishnamurthyRexford98] Balachander Krishnamurthy, Jennifer Rexford: "Software Issues in Characterizing *Web* Server Logs". Em artigo de opinião apresentado no World Wide *Web* Consortium Workshop on *Web* Characterization, Cambridge, Massachussets, Novembro, 1998, http://citeseer.nj.nec.com/krishnamurthy98software.html

[KristolMontulli00] D. Kristol, L. Montulli: "HTTP State Management Mechanism". *RFC 2965*, Outubro, 2000, http:// www.rfc-editor.org /rfc/rfc2965.txt

[Lombardi99] Michael Lombardi: "Computer Time Synchronization". 1999, http://tf.nist.gov/timefreq/service/pdf/computertime.pdf

[Lombardi01] Michael Lombardi: "Time and Frequency Measurements using the Global Positioning System (GPS)". *Cal Lab Magazine*, Julho-Setembro, 2001, http://tf.nist.gov/timefreq/general/pdf/1424.pdf

[Lombardi02] Michael Lombardi: "NIST Time and Frequency Services". *National Institute of Standards and Technology Special Publication 432*, 2002, http://tf.nist.gov/timefreq/general/pdf/1383.pdf

[Lourenço et al. 03] Anália Lourenço, Eurico Borges, Orlando Belo: "Utilização de Técnicas de Data WebHousing no Rastreio de Utilizadores e Análise de Dados de Sítios de Comércio Electrónico". Em *IV Conferência da Associação Portuguesa de Sistemas de Informação*, Universidade Portucalense, Porto, Portugal, 15-17 Outubro, 2003

[Luotonen95] A. Luotonen: "The Common Logfile Format". 1995, http://www.w3.org/pub/WWW/Daemon/User/Config/Logging.html

[MaxGeoIP] Maxmind, GeoIP Country, http://www.maxmind.com/download/geoip/database/GeoIPCountryCSV.zip

[MicrosoftCS2000] "Commerce Server Concepts". Documentação do *Microsoft Commerce Server 2000*, Microsoft Corporation, 2000

[MicrosoftKB296093] "Microsoft Knowledge Base Article – 296093: Prep*Web*Log Utility Prepares IIS Logs for SQL Bulk Insert". Microsoft Corporation, Maio, 2003, http://support.microsoft.com/default.aspx?scid=kb;EN-US; 296093

[MicrosoftKB318380] "Microsoft Knowledge Base Article – 318380: IIS Status Codes". Microsoft Corporation, Agosto, 2003, http://support.microsoft.com/default.aspx?scid=kb;en-us;318380

[MicrosoftIIS] Documentação online do IIS – Internet Information Services. Microsoft Corporation, http://www.microsoft.com/technet/prodtechnol/iis/default.asp

[MicrosoftIIS5RG00] "Microsoft Windows 2000 Server Resource Kits: Supplement 1 - Internet Information Services 5.0 Resource Guide". Microsoft Corporation, Microsoft Press, 2000

[MicrosoftIIS6RK03] "Internet Information Services (IIS) 6.0 Resource Kit", Microsoft Corporation, Microsoft Press, 2003

[MicrosoftIIS6RKT03] "Internet Information Services (IIS) 6.0 Resource Kit Tools". Microsoft Corporation, Maio, 2003, http://www.microsoft.com/downloads/details.aspx?familyid=56FC92EE-A71A-4C73-B628-ADE629C89499&displaylang=en

[MSSQLBooks04] "Data Transformation Services". *SQL Server 2000 Books Online*, Microsoft Corporation, 1998-2004

[Mills92] David L. Mills: "Network Time Protocol (Version 3) Specification, Implementation and Analysis". *RFC 1305*, Março, 1992, ftp://ftp.rfc-editor.org/in-notes/rfc1305.txt

[Mobasher et al. 01] Bamshad Mobasher, Bettina Berendt, Myra Spiliopoulou: "KDD for Personalization - PKDD 2001 Tutorial". Setembro, 2001, http://citeseer.nj.nec.com/mobasher01kdd.html

[MoeFader01] Wendy W. Moe, Peter S. Fader: "Capturing Evolving Visit Behavior in *Clickstream* Data". *Report 01-115*, Marketing Science Institute, 1000 Massachusetts Avenue, Cambridge, MA 02138, 2001 , http://fourps.wharton.upenn.edu/ideas/pdf/00-003.pdf

[Muehlen01] Michael zur Muehlen: "Process-driven Management Information Systems - Combining *Data Warehouses* and *Workflow* Technology". Em *Proceedings of the Fourth International Conference on Electronic Commerce Research (ICECR-4)*, Dalas, Texas, páginas 550-566, 8 a 11 de Novembro , 2001

[NCSAHTTPd] Documentação do servidor "NCSA HTTPd". NCSA, http://hoohoo.ncsa.uiuc.edu/docs/

[Netcraft03] Netcraft Ltd, "*Web* Server Survey". Setembro, 2003, http://news.netcraft.com/archives/*Web*_server_survey.html

[Netscape] Netscape Corporation, "Persistent Client State -- HTTP *Cookies*", http://www.netscape.com/newsref/std/*cookie*_spec.html

[Opera] Navegador Opera, Opera Software, http://www.opera.com

[OracleEBS] "Oracle e-Business Suite". Oracle Corporation, http://www.oracle.com/products/applications.html

[OWB04] "Oracle9i *Warehouse* Builder 9.2 Data Sheet". Oracle Corporation, 2004, http://otn.oracle.com/products/*Warehouse*/htdocs/datasheet92.htm

[Papa00] Johnny Papa: "Taming the Stateless Beast: Managing Session State Across Servers on a *Web* Farm", *MSDN Magazine*, Outubro, 2000

[PHC] PHC Software, http://www.phc.pt

[Primavera] Primavera Software, http://www.primaverasoftware.pt

[RahmDo00] Erhard Rahm, Hong Hai Do: "Data Cleaning: Problems and Current Approaches". *IEEE Bulletin of the Technical Committee on Data Engineering*, Volume 23, Número 4, páginas 3-13, 2000

[RealHelix03] "Helix Universal Server V9 - Administration Guide". Real Networks, Maio, 2003, http://service.real.com/help/library/servers.html#server

[SAPR3] SAP R/3 , SAP AG, http://www.sap.com

[SASWB02] "*WebHound*TM 4.1 Administrator's Guide". SAS Institute Inc., 2002

[SearchEng] "The Search Engine Watch". Jupitermedia Corporation, http://www.searchenginewatch.com

[Smith02] Diane Smith: "Oracle9iAS *Clickstream* Intelligence Administrator's Guide, Release 2 (9.0.2)", Maio, 2002

[Schafer et al. 01] J. B. Schafer, J. A. Konstan e J. Riedi: "e-Commerce Recommendation Applications". *Journal of Data Mining and Knowledge Discovery*, Springer-Verlag, volume 5, números 1/2, páginas 115-152, 2001

[SquidLog] "Squid *Log* Files", http://www.squid-*cache*.org/Doc/FAQ/FAQ-6.html

[Spiliopoulou et al. 03] Myra Spiliopoulou, Bamshad Mobasher, Bettina Berendt, Miki Nakagawa: "A framework for the evaluation of session reconstruction heuristics in *Web* usage analysis". *INFORMS Journal on Computing*, volume 15, número 2, páginas 171-190, Abril, 2003

[StJohns85] Mike St. Johns: "Authentication Server". *RFC 931*, Janeiro, 1985, http://www.rfc-editor.org/rfc/rfc931.txt

[StJohns93] Mike St. Johns: "Identification Protocol". *RFC 1413*, Fevereiro, 1993, http://www.rfc-editor.org/rfc/rfc1431.txt

[Sweiger et al. 02] Mark Sweiger, Mark R. Madsen, Jimmy Langston, Howard Lombard: "*Clickstream Data Warehousing*". John Wiley & Sons, 2002

[Unix2Dos] "Unix to Dos", http://www.bastet.com/software/UDDU.ZIP

[UnixUtil] "GNU utilities for Win32", http://unxutils.sourceforge.net/

[TanKumar02] Pang-Ning Tan, Vipin Kumar: "Discovery of *Web* Robot Sessions Based on their Navigational Patterns". *Journal of Data Mining and Knowledge Discovery*, Springer-Verlag, volume 6, número 1, páginas 9-35, 2002, http://www.cse.msu.edu/~ptan/Papers/DMKD.ps.gz

[W3C99] World Wide Web Committee Web Usage Characterization Activity: "W3C Working Draft: *Web* Characterization Terminology & Definitions Sheet". 1999, http://www.w3.org/1999/05/WCA-terms/

[Zaiane et al. 98] O. R. Zaiane, M. Xin, J. Han: "Discovering Web access patterns and trends by applying OLAP and data mining technology on Web logs". Em *Proceedings of Advances in Digital Libraries Conference (ADL98)*, páginas 19-29, Abril, 1998, http://citeseer.nj.nec.com/zaiane98discovering.html

[Zheng et al. 03] Zhiqiang Zheng, Balaji Padmanabhan, Steven O. Kimbrough: "On the existence and significance of data preprocessing biases in Web usage mining". *INFORMS Journal on Computing*, volume 15, número 2, páginas 148-170, Abril, 2003

Lista de Siglas e Acrónimos

ACL - *Access Control List*
ASP - *Active Server Pages*
BD - Base de Dados
CGI - *Common Gateway Interface*
COM - *Component Object Model*
CRM - *Customer Relationship Management*
DW - *Data Warehouse*
DM - *Data Mart*
DMZ - *Demilitarized zone*
DNS - *Domain Name Services*
DTS - *Data Transformation Services*
ER – Entidade-Relacionamento
ERM - *Electronic Relationship Management*
ERP – *Enterprise Resource Planning*
ETL - *Extraction, Transformation and Loading*
ETI - Extracção, Transformação e integração
FTP - *File Transfer Protocol*
GMT - *Greenwich Mean Time*
HOLAP - Hybrid OLAP
HTTP - *Hyper Text Transport Protocol*
HTTPS - *Secure Hyper Text Transport Protocol*
IIS - *Internet Information Services*

IP - *Internet Protocol*
LDAP - *Ligth-Weight Distributed Access Protocol*
MRP - *Manufacturing Resource Planning*
MOLAP - *Multidimensional OLAP*
NCSA - *National Center for Supercomputing Applications*
NFS - *Network File System*
NTP - *Network Time Protocol*
ODBC - *Open Database Connectivity*
OLTP - *On-line Transaction Processing*
OLAP - *On-line Analytical Processing*
RAID - *Redundant Array of Independent Disks*
RFC - *Request for Comments*
ROLAP - *Relational OLAP*
SGBD - Sistema de Gestão da Base de Dados
SO - Sistema Operativo
SFTP - *Secure* FTP
SSC - *Secure Copy*
SSL - *Secure Sockets Layer*
URI - *Universal/Uniform Resource Identificator*
URL - *Universal/Uniform Resource Locator*
UTC - *Coordinated Universal Time*
ZCD - Zona de Concentração de Dados

ANEXOS

ANEXO I Códigos de estado http

ANEXO II Códigos de estado e sub-estado HTTP específicos do servidor *Web* IIS

ANEXO III Estruturas de dados utilizada no *Webuts*

Anexo I - Códigos de estado http

A informação apresentada (Tabela A.1) é baseada na documentação do protocolo HTTP 1.1 apresentada no RFC2616 [Fielding et al. 99]. Serve apenas como guia rápido e não pretende substituir a consulta do dito RFC.

Tipo de código	***Valor***	***Descrição***
Informativo	100	Continue
	101	Mudando de protocolo
Sucesso	200	OK , pedido com sucesso
	201	Recurso criado
	202	Aceite para processamento
	203	Informação não obrigatoriamente credível
	204	Sem conteúdo
	205	Reconstruir conteúdo
	206	Conteúdo parcial
Redireccionamento	300	Escolhas múltiplas
	301	Mudou-se permanentemente
	302	Encontrado
	303	Consultar outro
	304	Não modificado
	305	Utilizar proxy
	306	(não utilizado mas está reservado na especificação)
	307	Redireccionamento temporário
Erros do cliente	400	Pedido inválido
	401	Sem autorização
	402	Pagamento necessário (código apenas reservado para futura especificação do protocolo)
	403	Não permitido
	404	Não encontrado
	405	Método não permitido
	406	Não aceitável
	407	Autorização pelo proxy necessária
	408	Pedido fora de prazo

	409	Conflito
	410	Suprimido
	411	Comprimento necessário
	412	Falha na pré-condição
	413	Recurso pedido demasiado grande
	414	URI pedido demasiado longo
	415	Tipo de dados não suportado
	416	Range (variável de cabeçalho) do pedido não pode ser satisfeito
	417	Expectativa falhada
Erros do Servidor	500	Erro interno do servidor
	501	Não implementado
	502	Gateway errado
	503	Servido não disponível
	504	Prazo de Gateway excedido
	505	Versão HTTP não suportada

Tabela A.1 – Códigos de estado do protocolo HTTP 1.1

Anexo II - Códigos de estado e sub-estado HTTP específicos do servidor *Web* Microsoft IIS

O servidor IIS da Microsoft proporcionada informação complementar ao código de estado HTTP definidos no RFC2616 [Fielding et al. 99]. Com a versão 6 do IIS este sub-estados (Tabela A.2) deixaram de ser transmitidos no cabeçalho HTTP ao cliente do pedido e passam a poder ser registados no campo `sc-substatus` [MicrosoftIIS6RK03] usado no formato *Microsoft IIS W3C Extended Log Format*.

Tipo de código	*Estado*	*Sub-estado*	*Descrição*
Informativo	100	N/A	Continue
	101	N/A	Mudando de protocolo
Sucesso	200	N/A	OK , pedido com sucesso
	201	N/A	Recurso criado
	202	N/A	Aceite para processamento
	203	N/A	Informação não obrigatoriamente credível
	204	N/A	Sem conteúdo
	205	N/A	Reconstruir conteúdo
	206	N/A	Conteúdo parcial
Redireccionamento	300	N/A	Escolhas múltiplas
	301	N/A	Mudou-se permanentemente
	302	N/A	Encontrado
	303	N/A	Consultar outro
	304	N/A	Não modificado
	305	N/A	Utilizar proxy
	306	N/A	(não utilizado mas está reservado na especificação)
	307	N/A	Redireccionamento temporário
Erros do cliente	400	N/A	Pedido inválido
	401	1	Sem autorização. Autenticação falhou
	401	2	Sem autorização. Autenticação falhou devido a configurações do servidor
	401	3	Sem autorização. Falta de autorização no Lista de controlo de acessos (ACL) do recurso

	401	4	Sem autorização. Autorização num filtro falhou
	401	5	Sem autorização. Autorização numa aplicação ISAPI/CGI falhou
	401	7	Sem autorização. Acesso negado resultante de uma política de autorizações de URL presente no servidor *Web* (IIS 6.0)
	402	N/A	Pagamento necessário (código apenas reservado para futura especificação do protocolo)
	403	1	Não permitido. Acesso para execução não permitido
	403	2	Não permitido. Acesso para leitura não permitido
	403	3	Não permitido. Acesso para escrita não permitido
	403	4	Não permitido. SSL necessário
	403	5	Não permitido. SSL 128 bits necessário
	403	6	Não permitido. Endereço IP rejeitado
	403	7	Não permitido. Certificado por parte do cliente necessário
	403	8	Não permitido. Acesso ao sítio negado
	403	9	Não permitido. Demasiados utilizadores
	403	10	Não permitido. Configuração inválida
	403	11	Não permitido. Alteração na chave de acesso
	403	12	Não permitido. Mapeador negou acesso
	403	13	Não permitido. Certificado do cliente revogado
	403	14	Não permitido. Listagem de directório negada
	403	15	Não permitido. Número de licenças de Access excedido
	403	16	Não permitido. Certificado do cliente é inválido ou de fonte não fiável
	403	17	Não permitido. Certificado do cliente expirou ou ainda não é válido
	403	18	Não permitido. Não pode executar URL pedido no actual contexto de aplicações (IIS 6.0)
	403	19	Não permitido. Não pode executar CGIs pedidos no actual contexto de aplicações (IIS 6.0)
	403	20	Não permitido. A autenticação pelo Passport falhou

			(IIS 6.0)
	404	N/A	Não encontrado
	405	N/A	Método não permitido
	406	N/A	Não aceitável
	407	N/A	Autorização pelo proxy necessária
	408	N/A	Pedido fora de prazo
	409	N/A	Conflito
	410	N/A	Suprimido
	411	N/A	Comprimento necessário
	412	N/A	Falha na pré-condição
	413	N/A	Recurso pedido demasiado grande
	414	N/A	URI pedido demasiado longo
	415	N/A	Tipo de dados não suportado
	416	N/A	Range (variável de cabeçalho) do pedido não pode ser satisfeito
	417	N/A	Expectativa falhada
Erros do Servidor	500	11	Erro interno do servidor. Aplicação está a terminar no servidor *Web*
	500	12	Erro interno do servidor. Aplicação está ocupada a reiniciar no servidor *Web*
	500	13	Erro interno do servidor. Servidor *Web* demasiado ocupado
	500	14	Erro interno do servidor. Configuração aplicacional no servidor inválida
	500	15	Erro interno do servidor. Pedidos directos ao ficheiro global.asa não são permitidos
	500	16	Erro interno do servidor. Credenciais de autorização do UNC inválidas (IIS 6.0)
	500	18	Erro interno do servidor. Armazenamento de autorizações URL não pode ser aberto (IIS 6.0)
	500	100	Erro interno do servidor. Erro interno de ASP
	501	N/A	Não implementado
	502	1	Gateway errado. Prazo de execução de CGI esgotado
	502	2	Gateway errado. Erro em aplicação CGI

	503	N/A	Servido não disponível
	504	N/A	Prazo de Gateway excedido
	505	N/A	Versão HTTP não suportada

Tabela A.2 – Códigos de estado e sub-estado HTTP fornecidos pelo servidor *Web* IIS

Anexo III – Estrutura de dados utilizada na ZCD do *Webuts*

Podemos ver nas ilustrações que se seguem os diagramas entidade-relacionamento da estrutura de dados que foi utilizada na Zona de Concentração de Dados do Webuts. As tabelas `redes_interna` e `motores_pesquisa` (Figura A.1) são as duas fontes de dados mantidas manualmente na ZCD.

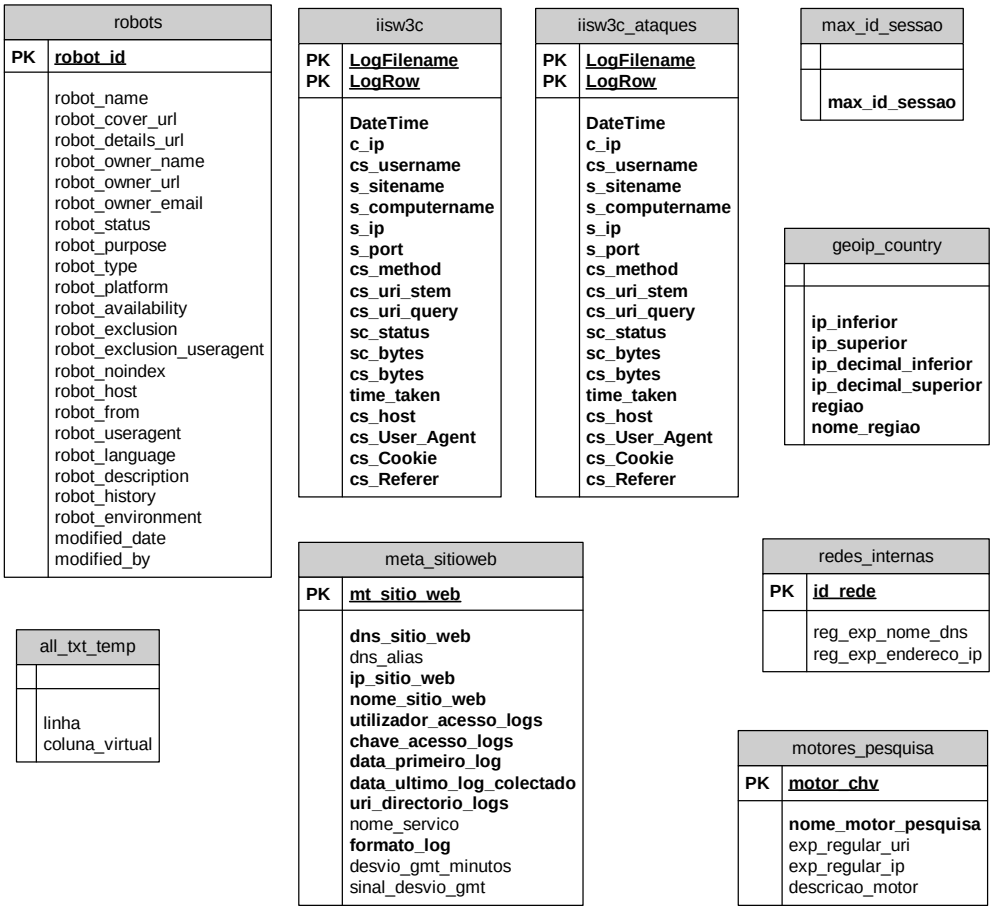

Figura A.1 – Tabelas utilizadas no carregamento e fontes de dados

As tabelas `robots`, `iisw3c`, `iisw3c_ataques`, `geoip_country` são utilizadas para como repositório dos dados colectados das fontes externas – *logs* do servidor *Web*, informação sobre robots e agentes automáticos e informação geográfica associada ao endereço IP. A tabela `mt_sitio_web` mantém informação de controlo e necessário para o funcionamento da colecta de ficheiros de *log*. A

tabela `max_id_sessao` armazena o valor de um contador usado para identificar univocamente as sessões.

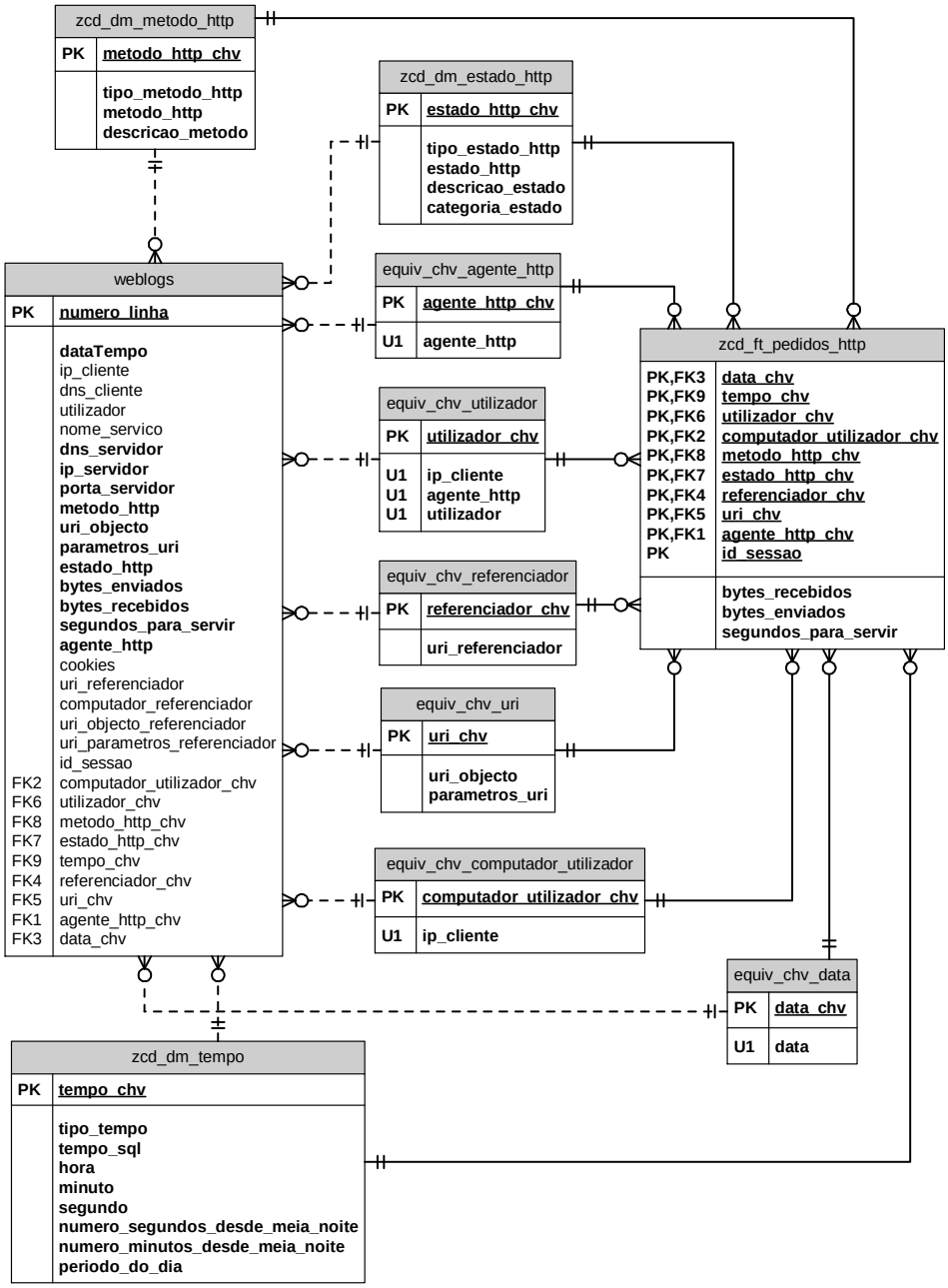

Figura A.2 – Equivalências e tabela de factos

A principal tabela para tratamento dos logs é a `weblogs` (Figura A.2). Será aqui onde os processos de geração das dimensões irão colocar a chave de substituição que mais tarde irá ser necessária para formação da tabela de factos. As tabelas para armazenamento temporário dos valores das dimensões na ZCD começam com o prefixo `ZCD_DM`. Tanto estas como a tabela de factos são criadas na ZCD à imagem das existentes no sistema operacional da DW.

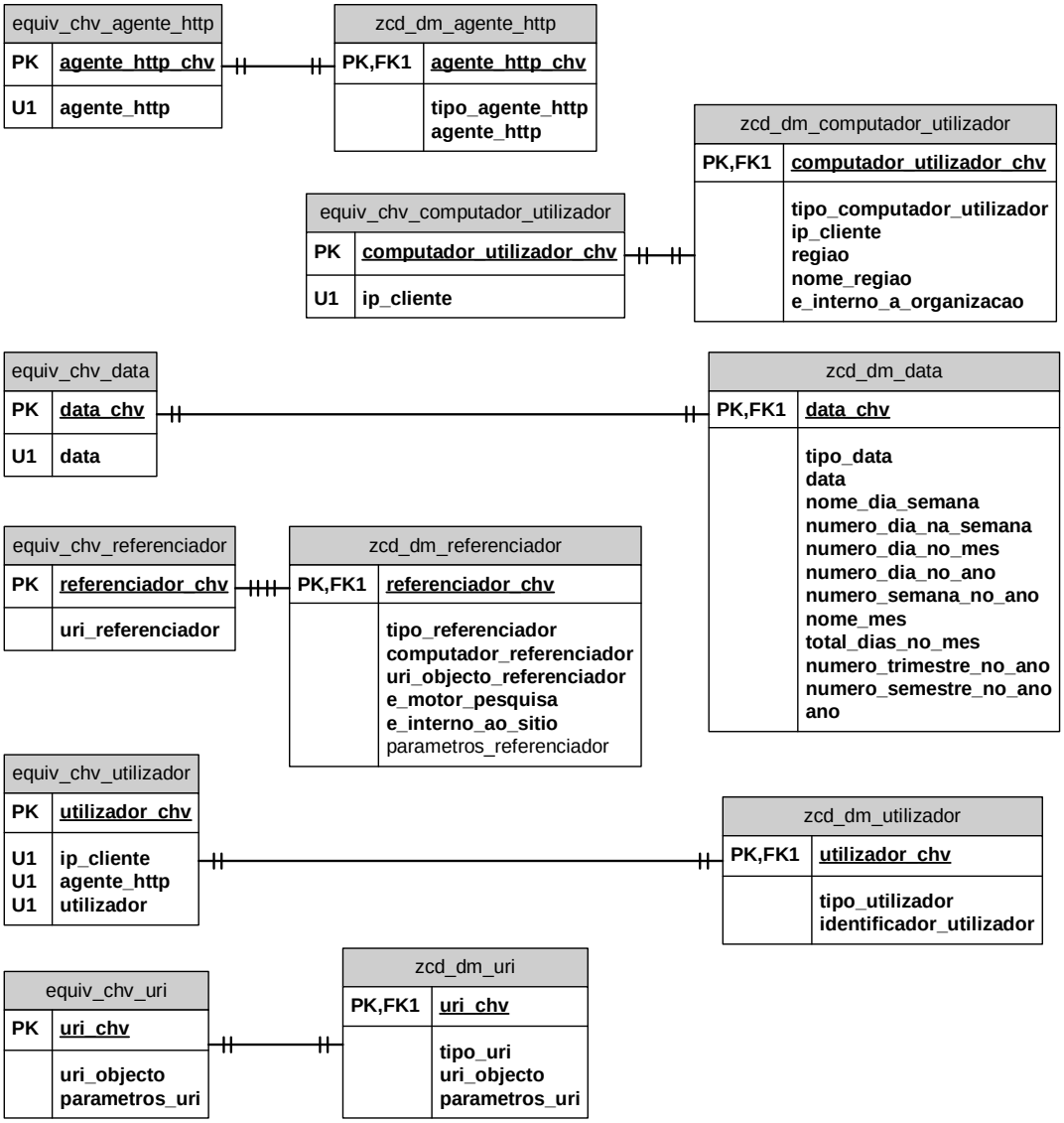

Figura A.3 – Tabelas de armazenamento de chaves de substituição

As tabelas com o prefixo `equiv_` (Figura A.3) são utilizadas para armazenamento e equivalências das chaves de substituição. Pode-se notar que para as dimensões Tempo, Método http e Estado http não existem estas tabelas de equivalências pelo facto de existir permanentemente uma cópia dessas dimensões mantida na Zona de Concentração de Dados.