UNIVERSIDADE DE LISBOA

FACULDADE DE CIÊNCIAS

DEPARTAMENTO DE INFORMÁTICA

BUSINESS INTELLIGENCE EM APLICAÇÕES FCUL

Ricardo Jorge da Costa Simões

MESTRADO EM INFORMÁTICA

Trabalho de Projeto orientado por:
Prof. Doutor João Pedro Guerreiro Neto
e Licenciado Rui Miguel Barata Nunes

2016

Agradecimentos

Aproveito esta oportunidade para agradecer a todos aqueles que possibilitaram a conclusão deste projeto.

Aos meus colegas da Unidade de Informática da Faculdade de Ciências especialmente ao meu coorientador Rui Nunes.

Ao meu orientador Professor Doutor João Neto.

Aos meus familiares e amigos.

Resumo

O *Business Intelligence* (BI), cada vez mais em voga nos tempos que correm, é um conjunto de técnicas e ferramentas que usa os dados produzidos pelos sistemas operacionais de uma empresa ou instituição e transforma-os em informação útil para os seus decisores. Essa informação é obtida e armazenada periodicamente, permitindo uma análise comparativa através do tempo podendo-se produzir relatórios, *data mining*, aplicar cubo de dados, entre outros.

A Unidade de Informática (UI) da Faculdade de Ciências da Universidade de Lisboa (FCUL), para além de garantir o funcionamento de toda a infraestrutura de *hardware* (rede e computadores), fornece também um leque variado de serviços aos alunos, docentes e funcionários. Grande parte desses serviços estão no portal *web* da FCUL (http://www.ciencias.ulisboa.pt). Sendo a informática algo transversal a todos os departamentos e permitindo o normal funcionamento da faculdade, é fundamental que os serviços prestados pela unidade de informática sejam os mais eficientes possíveis, que cheguem a um maior número de utilizadores e que tenham um maior número de funcionalidades ao dispor.

Com o objetivo de melhorar a qualidade dos serviços prestados, fazer um maior aproveitamento dos recursos e apresentar evidências aos outros departamentos de métricas de desempenho, a UI pretende o desenvolvimento de uma *data warehouse* que reúna toda a informação que seja relevante de uma forma coerente e acessível. Essa informação deverá permitir fazer uma análise exploratória combinando vários critérios de filtragem e gerar *reports* periódicos.

Para finalizar, esta *data warehouse* deverá ser desenvolvida preferencialmente com tecnologias *open source* e ser facilmente escalável a novos *data marts*

Palavras-chave:

Data Mart – Data Warehouse - Business Intelligence - Data Mining – open source

Abstract

Business Intelligence (BI) is a set of techniques and tools that use the raw data produced by operational systems of a company or institution and transforms them into useful information for decision makers. This information is periodically obtained and stored enabling a comparative analysis over time. With this stored information you can produce reports, data mining, data cubes, etc.

The Unidade de Informática (IT Department) of the Faculty of Sciences, University of Lisbon (FCUL) guarantees the functioning of all the hardware infrastructure (network and computers). It also provides a range of services to students, teachers, and staff. Many of those services are located in the faculty portal (http://www.ciencias.ulisboa.pt). Being used in all departments and allowing their normal functioning is essential that the services provided by IT Department be the most efficient possible, reach the greatest possible number of users and have a greater number of features available.

In order to improve the quality of services, making better use of resources and presenting evidences to other department's performance and business metrics, the IT Unit intends to build and develop a data warehouse bringing together all relevant information in a coherent and accessible way. This information should generate regular reports, allowing an exploratory analysis combining several filters.

Finally, this data warehouse should be developed preferably with open source technologies and should be easily scalable to new data marts.

Keywords:

Data Mart – Data Warehouse - Business Intelligence - Data Mining – open source

Conteúdo

Glossário

Active Directory

Serviço de diretório no protocolo LDAP que armazena informações sobre objetos pertencentes a uma rede de computadores como por exemplo computadores, impressoras ou utilizadores e disponibiliza essas informações a usuários e administradores desta rede.

Auditing

Processo de auditoria que regista as operações efetuadas por um utilizador.

Batch

Processamento em lote que executa várias tarefas sequencialmente.

Big Data

É um termo utilizado para nomear conjuntos de dados digitais muito grandes ou complexos, que os aplicativos de processamento de dados tradicionais ainda não conseguem lidar com muita eficácia.

Bus Matrix

Ferramenta utilizada para criar, documentar e comunicar a arquitetura da *data warehouse*.

Business Intelligence

Processo tecnológico para analisar e apresentar informação que possa auxiliar na tomada de decisão em organizações.

CakePHP

Framework Model View Controller para PHP.

Client Machine

Características do computador do utilizador que acede a um determinado servidor.

Cloud

Utilização da memória e da capacidade de armazenamento e cálculo de computadores e servidores compartilhados e interligados por meio de uma rede externa ou interna.

Clustering

Técnica estatística usada no *data mining* para fazer agrupamentos automáticos de dados segundo seu grau de semelhança.

CMS Drupal

Gestor de conteúdos implementado em PHP.

Cron

Comando de Unix / Linux que possibilita executar uma tarefa periodicamente.

Cubo OLAP

Um cubo OLAP é um *array* multi-dimensional em que se pode efectuar operações de *slice & dice*, *drill down*, *roll-up*, etc.

Data flow

Aplicado ao ETL determina o caminho que os dados podem tomar a nível de transformações e processamento.

Data Marts

Aplicação que colige um subconjunto de informação de natureza analítica e relevante para um grupo de utilizadores.

Data Mining

Conjunto de técnicas que permitem explorar grandes quantidades de dados à procura de padrões e relações sistemáticas entre as suas variáveis.

Data warehouse

Sistema que suporta uma base de dados separada e dedicada ao suporte à decisão. Os dados são transferidos dos sistemas operacionais e outros sistemas externos e integrados no novo sistema de natureza analítica.

Deployed

Instalado nos servidores.

Drill across

Mecanismo de análise que permite navegar horizontalmente por entre vários *data marts*.

Drill down

Mecanismo de análise que permite navegar ao longo dos níveis de uma estrutura multidimensional, partindo do global para o detalhe.

Endpoint

Endereço de um serviço REST que ao ser invocado efetua uma determinada ação.

Fact Constellation

Diagrama em estrela conjugando as várias tabelas de factos de uma *data warehouse*.

LOGOS

Sistema de informação da FCUL que guarda toda a informação da faculdade relativa a docentes, funcionários, publicações, etc.

Medidas

Valores que estão associados a uma determinada ação na tabela de factos. Por exemplo: valor monetário ou tempo gasto, etc.

Microsoft SSAS

Microsoft Analysis Services é um produto da Microsoft que tem um conjunto de funcionalidades para gerir *data warehouse* através de cubos OLAP.

Microsoft SSIS

SQL *Server Integration Services* é uma ferramenta da Microsoft de integração de dados e ETL.

Middleware

Aplicação que medeia a informação entre aplicações. É utilizado para mover ou transportar informações e dados entre programas de diferentes protocolos de comunicação, plataformas e dependências dos sistemas operativos.

Model View Controller

Paradigma de desenvolvimento de *software* que separa as camadas de apresentação, dos dados e do controle desses dados para serem apresentados.

Moodle

Plataforma académica open *source desenvolvida* em PHP

Open Source

Software que possui o seu código fonte aberto e gratuito podendo ser utilizado e modificado.

Outliers

Termo estatístico referente a um valor atípico fora do que é normal.

Parsing

Processo de analisar uma sequência de caracteres retirando-lhe informação útil.

Pivoting

Rotação do cubo de dados em torno de um eixo.

Presentation area

Área onde são guardados os dados da *data warehouse* acedidos pelo cubo OLAP e por sua vez pelos utilizadores.

Reverse engineering

Inversão do processo de desenvolvimento, partindo do produto final para as partes que o constituem.

Role-playing

Na modelação dimencional, uma mesma dimensão pode assumir papéis diferentes. Por exemplo se um facto tem uma data de início e uma de fim, não faz sentido existirem duas tabelas que representam exactamente a mesma informação.

Roll-up

Operação inversa do *drill-down* em que se parte do maior detalhe para a agregação dos dados.

Server-side

Ações que ocorrem no lado do servidor.

Servlet

Classe JAVA usada para extender as funcionalidades de um servidor, geralmente servidores web.

Serviços REST

Representational State Transfer (REST) é uma abstração de arquitetura de sistemas que segue um conjunto de regras de boas praticas como o uso de hipermédia, sem estado, etc.

SIGES

Sistema de informação da FCUL que guarda a informação referente aos alunos.

Slice & dice

Slice é o ato de restringir num subconjunto do cubo OLAP escolhendo um valor fixo de uma das dimensões. *Dice* é uma operação que produz um sub cubo escolhendo um conjunto de valores de várias dimensões.

Snapshots

Estado de um sistema num determinado momento.

Snowflaking

Modelo de dados que difere do esquema em Estrela (ver *Star schema*) ao nível de normalização. As tabelas de dimensão de um esquema *snowflake* são geralmente normalizadas pela terceira forma de normalização (3NF).

Staging area

Área intermédia entre os sistemas operacionais e a *presentation area* onde os dados são tratados e preparados para serem carregados posteriormente para a *presentation area*.

Star schema

Diagrama do modelo de dados dimensional que num núcleo central, composto por dados, ou medidas, rodeado por várias tabelas com descritores que caraterizam factos.

Surrugate key

Chave substituta que identifica univocamente um elemento de uma dimensão no *data warehouse.*

Tomcat

Apache Tomcat é um servidor aplicacional de JAVA.

Tuning

Processo de afinação.

User Agent

Informação que o *browser* fornece ao servidor sobre a máquina cliente.

Weka

Framework de algoritmos de *data mining*

Acrónimos e abreviaturas

Termo	Descrição
3NF	Third normal form
AD	Active Directory
ARFF	Attribute-Relation File Format
BD	Base de dados
BI	Business Intelligence
DW	Data warehouse
DM	Data mart
ETL	Extration Transfransform Load
FCUL	Faculdade de Ciências da Universidade de Lisboa
GLPI	Gestion libre de parc informatique
JDBC	Java Database Connectivity
MDX	Multidimensional Expressions
MVC	Model View Controller
OLAP	Online Analytical Processing
OLTP	Online Transaction Processing
REST	Representational State Transfer
UI	Unidade de Informática
URL	Uniform Resource Locator
XMLA	XML for Analysis

Lista de Figuras

Lista de Tabelas

Capítulo 1 Introdução

A Faculdade de Ciências da Universidade de Lisboa (FCUL) ocupa treze edifícios com serviços centrais, departamentos das várias áreas do conhecimento, laboratórios e centros de investigação, biblioteca, salas de aula e espaço de estudante.

A Unidade de Informática (UI), para além de garantir o funcionamento de toda a infraestrutura de *hardware* (rede, *wi-fi* e computadores), fornece também um leque variado de serviços, tais como aplicações para docentes e alunos para auxiliar no leccionamento dos cursos, aplicações para os funcionários internos da faculdade, serviços de e-*mail*, filmagens e multimédia entre outros. A UI está dividida pelos núcleos:

O **Núcleo de Sistemas de Informação e Desenvolvimento** desenvolve aplicações *ad hoc* para outras unidades funcionais da faculdade e mantém as bases de dados da Unidade Académica e Funcionários.

O **Núcleo de Suporte a Utilizadores**, **E-learning e Multimédia** tem como funções coordenar as ações dos operadores do *front office*, manter o *Moodle*, feramentas de *e-learning* e multimédia (filmagens e edição de imagem).

O **Núcleo de Infraestrutura de Serviços e Servidores** realiza a administração de sistemas dos servidores da FCUL de *e-mail*, *web*, base de dados, *active directory*, *proxys*, *domain controllers*, etc.

O **Núcleo de Infraestrutura de Comunicações** gere a infraestrutura de rede dos edifícios da faculdade, a rede *wi-fi* e a *firewall*.

Existem quatro tipos de utilizadores que podem interagir com a UI e com estes núcleos:

- **Funcionário:** Possui um vínculo laboral com a faculdade e depende da infraestrutura de rede, *software* e computadores para realizar o seu trabalho no dia-a-dia.

- **Docente:** Leciona na faculdade possuindo as mesmas necessidades de um funcionário e ainda a necessidade de alocação de salas de aulas, configuração de computadores em laboratórios, lançamento de notas, etc.
- **Aluno:** Tal como os funcionários e docentes também utiliza os recursos informáticos da FCUL, mas numa forma mais limitada.
- **Anónimo:** Utilizador que acede ao portal da FCUL, mas que não efetuou *login.*

Para além dos núcleos, a UI é composta por um *front office* aberto ao público com operadores e que funciona como primeira linha onde os alunos, docentes e funcionários efetuam pedidos. Existe um *back office* de técnicos de cada área dos núcleos que funciona como 2ª linha. Para a interação, entre o utilizador, a 1ª linha e a 2ª linha, existem aplicações *web* para gerir os diversos tipos de pedidos. Estes pedidos vão consumir tempo a um ou mais operadores e técnicos e gerar acompanhamentos.

1.1 Motivação

Muitas vezes a UI é confrontada com a necessidade de obter indicadores para justificar decisões perante os departamentos ou o conselho diretivo. É possível extrair esses indicadores diretamente dos sistemas operacionais, mas implica construir *queries* à medida que costumam ser pesadas a nível de processamento. As aplicações que a UI disponibiliza são orientadas a serviços muito específicos e não existe uma vista unificada sendo por vezes necessário cruzar várias origens de dados. O estado destes dados varia ao longo do tempo e não existe um histórico que permitiria perceber a evolução / retrocesso de certos indicadores. Também é importante compreender o tempo que todos os operadores e técnicos despendem e em que tarefas.

Para solucionar estas e outras questões procede-se à construção de um sistema de apoio à decisão OLAP. Este sistema é paralelo e independente aos sistemas operacionais, não causando degradação de desempenho. A construção de uma *data warehouse* permite reunir num só local várias fontes de dados. Estas fontes podem ser tratadas eliminando erros, normalizando dados e resolvendo sobreposições. Neste sistema centralizado, o modelo de dados é bastante mais simplificado permitindo uma análise exploratória dos dados mais simples e intuitiva.

Sendo a área de BI cada vez mais importante para as empresas ou instituições na obtenção de informação analítica para os decisores - que em condições normais não seria acessível ou inteligível - identificou-se a necessidade de aplicar, através do

aprofundar dos conhecimentos nesta área, as metodologias necessárias, pretendendo-se que a médio-longo prazo os resultados desta *data warehouse* sejam úteis para a melhoria dos serviços prestados pela UI à FCUL

Diariamente a UI é confrontada com a necessidade de avaliar e apresentar frequentemente vários indicadores. Os mais importantes têm a ver com o tempo despendido dos operadores e técnicos nas várias tarefas. As tarefas que possam consumir mais tempo devem ser otimizadas de modo a libertar recursos para outras tarefas. Outro indicador é o número de acompanhamentos de uma determinada tarefa, ou seja, a troca de mensagens e o número de pessoas envolvidas para a resolver.

Outra utilidade na construção deste sistema de DW / BI será conhecer melhor, de uma forma estatística, os utilizadores que interagem com a UI. Se são mais alunos ou docentes; nacionais ou estrageiros; qual a sua idade; etc. Todos estes indicadores devem produzir relatórios facultados periodicamente aos departamentos e ao conselho diretivo. Juntam-se a estas necessidades informativas, uma possível análise de padrões de utilização dos serviços da UI recorrendo a *data mining.*

1.2 Objetivos

O objetivo para a construção de uma *data warehouse* são:

- Simplificar o acesso aos dados.
- Ser o mais transversal possível de modo a acomodar um maior número de áreas de negócio.
- Ser capaz de lidar com várias fontes de dados (*logs* de servidores, base de dados relacionais, e outras eventuais fontes)
- Guardar dados coerentes.
- Eliminar erros e redundâncias.
- Fiabilidade dos dados.
- Adaptação às mudanças.
- Permitir de futuro poder alterar os atributos da *data warehouse*.
- Permitir expandir e criar novos *data marts.*
- Melhorar os processos de decisão.
- Produção de vários tipos de relatórios que facilitem o apoio à decisão.
- Ser desenvolvido preferencialmente com ferramentas *open source.*

- Escalável e de fácil manutenção.
- Permitir a eliminação de logs libertando recursos.
- Gerar *reports* mensais:
    - Para cada departamento
    - Interno à UI
    - Público
- Haver ferramentas que permitam gerar relatórios dinâmicos.

Os objetivos pessoais na elaboração deste projeto são:

- Aprofundar os conhecimentos na construção de uma *data warehouse* e de *data mining* através de um caso de estudo real.
- Descobrir e utilizar tecnologias open source que permitam a construção da DW.
- Comparar essas tecnologias open source com as ferramentas da Microsoft SSIS e SSAS
- Aprender a produzir um relatório e documento deste género.
- Aprofundar os conhecimentos das aplicações *web* da FCUL.

1.3 Organização do documento

Este documento terá uma lista de capítulos iniciais referentes ao levantamento de requisitos e sua análise, uma lista de capítulos referentes à implementação da *data warehouse* e finalmente capítulos referentes à análise dos dados obtidos pela DW e conclusões.

Nos capítulos relativos à análise temos o capítulo dois referente à metodologia e planeamento, o capítulo três referente ao levantamento de requisitos; o capítulo quatro referente ao desenho da arquitetura e escolha dos produtos; o capítulo cinco referente aos processos de negócio e finalmente o capítulo seis referente à modelação dimensional.

Para os capítulos de desenvolvimento temos o capítulo sete referente ao modelo físico; o capítulo oito referente ao processo de ETL; o capítulo nove referente à definição do cubo de dados; o capítulo dez referente às ferramentas analíticas e o capítulo onze referente ao *data mining*.

Na análise dos dados temos o capítulo doze referente a resultados e respostas às perguntas analíticas e a conclusão.

Existem ainda os anexos relativos à análise dos dados dos sistemas operacionais, lista de ferramentas para BI, estrutura do ficheiro arff para weka, *output* dos algoritmos

de classificação do weka, schema do Mondrian e formatação do tempo gasto numa *query* de MDX.

Capítulo 2 Metodologia e planeamento

Uma implementação bem sucedida de um DW / BI depende da correta integração entre as várias tarefas e componentes. Tem como objetivo fazer uma gestão de esforço dos recursos, prever e mitigar eventuais problemas, estimar e assegurar metas e datas de entrega e assegurar a qualidade do produto final.

Devido ao facto de ser difícil de explicar o que é uma *data warehouse* aos utilizadores, decidiu-se enveredar pela abordagem *top down*, em que se escolhe e se implementa um *data mart* inicial e depois se expande a DW implementando os outros. Esta abordagem permite por um lado mostrar aos intervenientes da UI uma prova de conceito que ajuda a esclarecer da pertinência e utilidade de ter um sistema DW / BI e por outro será um teste para as opções efetuadas ao nível de arquitetura e de escolha de tecnologias.

A metodologia a aplicar será a proposta por Ralph Kimball [1,2,3]:

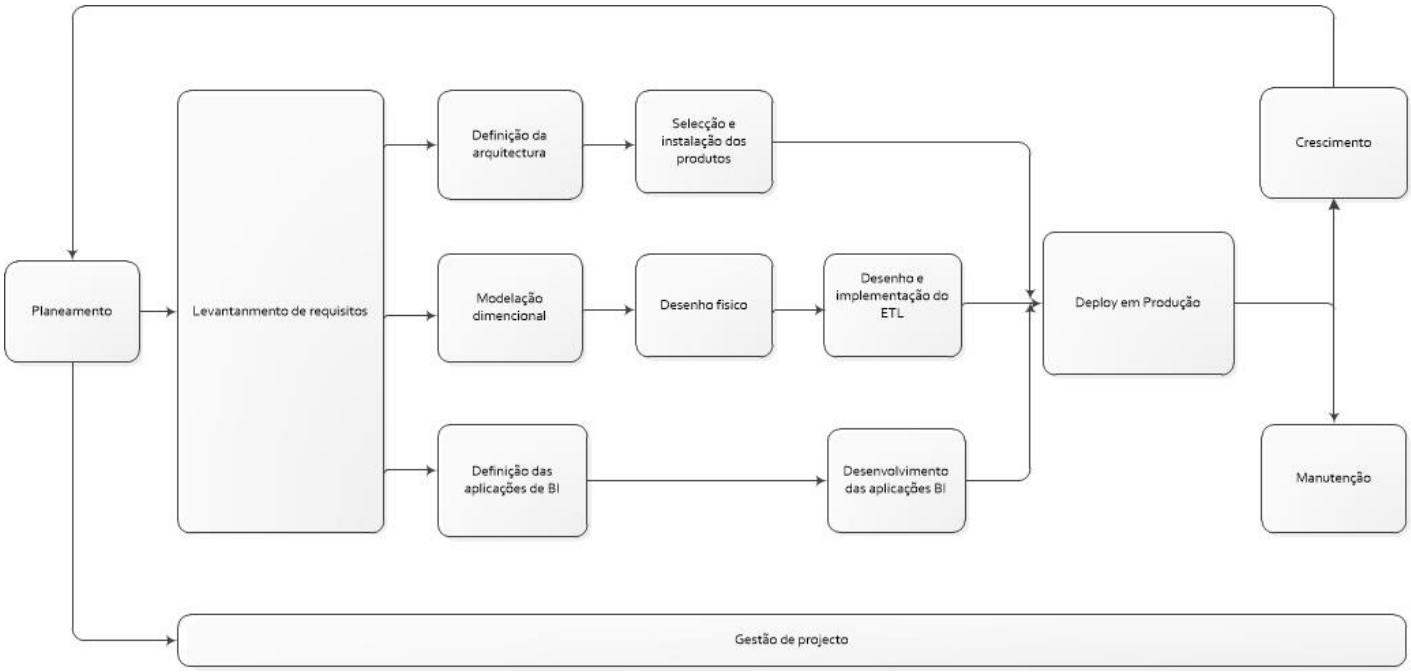


Figura 1- Metodologia de Kimball

As tarefas são as seguintes:

Planeamento:

É necessário entender os requisitos do sistema, identificar tarefas, a sua duração e as suas dependências.

Gestão de projeto:

Tem como finalidade a coordenação de recursos, infraestruturas e comunicação entre equipas. Monitoriza a evolução do projeto e faz a ponte entre os elementos do negócio das diversas áreas com o desenvolvimento.

Levantamentos de requisitos:

Perceber o que os utilizadores pretendem. Estabelecer os requisitos de negócio, funcionais e não funcionais. Fazer uma análise nas fontes de dados.

Deve proceder a uma compreensão das diversas áreas de negócio e as suas necessidades. Permitir definir quais os *data marts* a serem criados, a sua prioridade e as perguntas analíticas associadas. Determinar também quais os requisitos / restrições tecnológicas derivadas da arquitetura dos sistemas operacionais e das políticas da UI.

Definição da arquitetura:

É necessária a integração de várias tecnologias. Três fatores devem ser tomados em conta: requisitos de negócio, origem dos dados dos sistemas operacionais e filosofias / estratégias da UI referentes a tecnologias. É preciso resistir à tendência natural de focar exclusivamente na parte tecnológica.

Seleção e instalação dos produtos:

Escolher os produtos para o ETL, cubo de dados, ferramentas analíticas, *data mining*, ponderando o que foi definido na arquitetura e as que podem melhor servir os utilizadores e os requisitos.

Conhecendo as restrições tecnologias dos OLTP, as suas regras e políticas, elabora-se a arquitetura, a escolha das ferramentas e dos produtos adequados a cada fase: Para analisar as fontes dos dados, construir o ETL, construir o Cubo de dados e ferramentas analíticas.

Modelação dimensional:

Através do levantamento de requisitos definir quais são os *data marts* e para cada um determinar quais são as dimensões e as hierarquias dos seus atributos, o que são os factos, a sua granularidade, etc. Definir o *bus matrix* da *data warehouse*. Esta matriz será a fundação para futuras integrações de dados no futuro porque associa as dimensões usadas por cada *data mart*.

Conhecendo os processos de negócio procede-se à modelação dimensional proposta por *Kimball*, em que se construirá um modelo em estrela (*star schema*), onde existem várias dimensões e uma tabela de factos com as chaves estrangeiras dessas dimensões e as diversas medidas.

Desenho físico:

Processo que faz a passagem da modelação dimensional para a sua implementação numa base de dados relacional. Esta não é necessariamente um decalque das tabelas da modelação dimensional, visto que muitas vezes existem mais campos e tabelas auxiliares que ajudam no processo de ETL. Existem dois modelos físicos, um para a *staging area* e outro para a *presentation area*. A *staging area* é onde o ETL é efetuado e a *presentation area* onde os dados tratados são colocados e acedidos pelo cubo de OLAP (consultar o capitulo 7 – Desenho Físico).

ETL:

Desenho e implementação da extração, transformação e carregamento dos dados. Este é talvez o processo mais crucial e o que toma mais tempo. Este processo consiste em aceder às várias fontes de dados, extraindo a informação relevante, eliminando erros e redundâncias e carregando na área de apresentação dos dados.

Definição das aplicações de BI:

Decide-se por enveredar pela implementação de raiz ou recorrer a aplicações existentes no mercado. Quer num caso, quer outro, é preciso adaptar essas aplicações às necessidades dos utilizadores finais quer a nível de estrutura dos dados, quer a nível de vocabulário usado.

Desenvolvimento das aplicações de BI:

As aplicações são implementadas ou configuradas de acordo com o que foi definido no processo “definição de aplicações BI”.

Deploy:

O sistema é colocado em produção. É necessário definir a operacionalização do ETL visto que este deverá ser executado periodicamente. Para tal é preciso ver a periodicidade a que os utilizadores precisam de dados atualizados, o tempo que demora o processo de ETL e a melhor altura de execução de modo a interferir o menos possível com os sistemas operacionais.

Manutenção:

Estando o sistema em produção é preciso monitorizá-lo para ver se continua com bom desempenho recorrendo a *tuning* de BD (índices, etc.). Pode também ser necessária a intervenção manual em algum processo que não tenha sido possível implementar no ETL.

Crescimento:

O crescimento implica acompanhar a evolução da instituição ou empresa, refletindo as alterações no DW. Essas alterações podem ser novos atributos nas dimensões, novas dimensões ou até mesmo novos *data marts.*

Capítulo 3 Levantamento de requisitos

O levantamento de requisitos na construção de uma *data warehouse*, tal como em todos os processos de desenvolvimento de *software*, é uma fase bastante importante. Nesta fase vai-se compreender os requisitos dos utilizadores e as regras de negócio. Para fazer o levantamento desses requisitos, a primeira tarefa consiste em entrevistar os intervenientes que vão ser os utilizadores finais da solução DW / BI. Estas entrevistas permitem:

- **Conhecer as funções desempenhadas pelos utilizadores:** Conhecendo o seu trabalho no dia-a-dia vai permitir compreender como uma solução DW / BI os pode ajudar a melhorar o seu desempenho e os auxiliar no apoio à decisão.
- **Conhecer o seu modo de ver a organização e vocabulário:** A DW tem que ser fácil e intuitiva de usar, estar organizada e conter vocabulário que os utilizadores estejam habituadas a usar.
- **Conhecer as aplicações dos sistemas operacionais:** É importante conhecer as aplicações que vão servir como fonte de dados para a DW e quais são os seus aspetos mais relevantes que possam ser traduzidas na modelação dimensional, ou seja, as entidades inerentes no negócio que vão das origem a dimensões, os seus campos e as ações que vão ser traduzidas em factos.
- **Compreender as espectativas face ao DW:** Perceber como os utilizadores veem a utilidade de um sistema DW / BI, que funcionalidades gostariam que fossem implementadas e outros requisitos não funcionais tais como performance, disponibilidade, etc.

Para além das entrevistas, o levantamento de requisitos foi feito através de uma análise dos dados dos sistemas operacionais (ver Anexo A - Analise da origem dos dados do OLTP), leitura de documentação e *reverse engineering* de algumas aplicações.

Existem requisitos globais que são transversais (ver secção 3.1) a todos os núcleos da UI e outros que são específicos (ver secções 3.2 – 3.5).

3.1 Requisitos transversais

3.1.1 Requisitos funcionais

- A ferramenta de ETL ser fácil de configurar e de alterar.
- O processo de ETL deve correr periodicamente e de forma automática.
- Existir um cubo OLAP entre os dados da DW e as ferramentas analíticas.
- Haver a possibilidade de utilizar mais do que uma ferramenta analítica.
- As ferramentas analíticas devem permitir gerar listas e gráficos.
- As ferramentas analíticas devem permitir exportar dados para Excel.
- Produção de *reports* em formato PDF para serem enviados via correio eletrónico.

3.1.2 Requisitos não funcionais

- As ferramentas analíticas devem ser intuitivas, permitindo utilização por qualquer operador ou técnico da UI.
- As tecnologias utilizadas na construção deste DW devem ser *open source* e não imputarem custos à UI.
- O DW deve apresentar dados coerentes e credíveis.
- As várias origens de dados devem estar integradas para que não hajam incoerências, redundâncias que ponham em risco a fiabilidade da informação obtida.
- As tecnologias utilizadas devem ser acessíveis aos programadores do núcleo de desenvolvimento da UI.
- Os dados apresentados e nomenclatura existentes em todo o DW devem ser compreensíveis por todos os funcionários da FCUL.
- O acesso aos dados da DW devem ser seguros e haver políticas de acesso.
- Toda a informação acerca dos utilizadores existente no DW deve-se cingir ao essencial.
- A DW deve-se adaptar a mudanças: necessidades dos utilizadores, necessidades de negócio, novas origens de dados ou tecnologias.
- Os dados da DW devem ser apresentados em tempo útil: à medida que os sistemas operacionais vão criando novos dados em bruto para serem introduzidos no DW, este processo deverá decorrer num espaço de tempo

útil que não ponha em risco as necessidades dos utilizadores para aceder a esses novos dados.

3.2 Requisitos para o Núcleo de Suporte a Utilizadores, E-learning e Multimédia

3.2.1 Funções desempenhadas e necessidades negócio

Os utilizadores podem interagir através de vários **canais** tais como via *web*, telefone, *email* ou de forma presencial. De todas as formas, essa interação será registada no sistema, indicando o **utilizador** em questão, a natureza da **operação**, quem a tratou e o **tempo gasto**. Existem operações que dada a sua complexidade podem ter que ser resolvidas por mais do que um **operador / técnico**. A esse número de interações chama-se **acompanhamentos**. É importante saber a **data** e **hora** em que essa interação foi feita bem como o **estado** em que se encontra.

Para além de aferir o tempo gasto, é importante a este núcleo conhecer melhor os utilizadores e que tipo de operações mais comuns costumam efetuar.

3.2.2 Origem dos dados

Para desempenhar as tarefas do dia-a-dia, este núcleo dispõe das seguintes aplicações que vão ser as suas fontes de dados: “Gestão de Pedidos”, “Inbox”, “Gestão de tomadas”, “Gestão de grupos” e nova aplicação de “gestão de pedidos GLPI”. De referir que durante a construção deste *data warehouse*, a aplicação “gestão de pedidos” e “inbox” desenvolvidas pela UI foram descontinuadas e substituídas por esta aplicação *open source* GLPI.

Gestão de pedidos

Esta aplicação permite registar as interações que os utilizadores têm com a UI. Os utilizadores têm vários canais por onde podem requerer: Presencial, Telefone, Correio Eletrónico e *Web*.

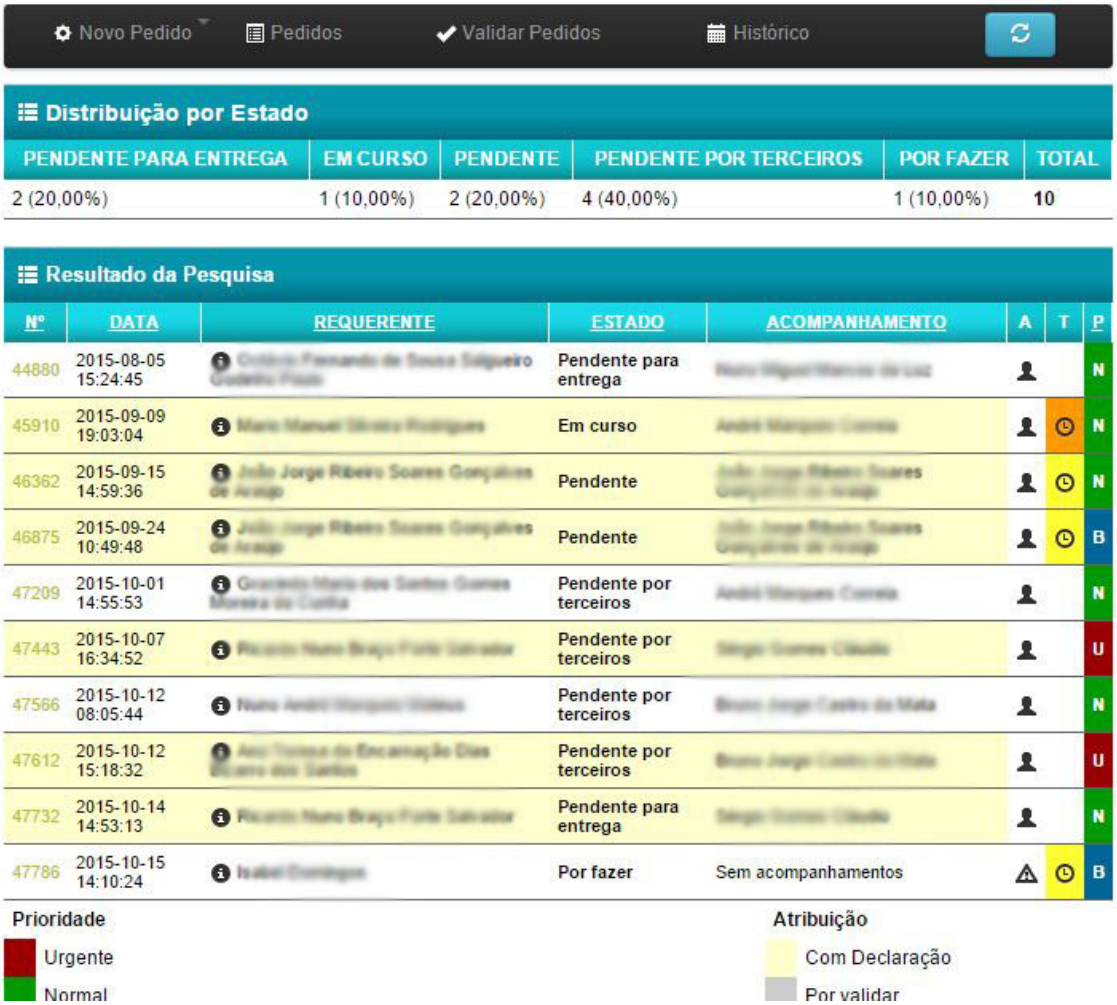


Figura 2 – Back office da aplicação "Gestão de pedidos"

Independentemente do canal, os operadores têm que introduzir na aplicação "Gestão de pedidos" o pedido que foi efetuado e o tempo gasto, excetuando quando o pedido é efetuado pelo próprio utilizador via web.

Existem várias naturezas de pedido: problemas de *hardware*, instalação de *software*, tomadas de rede e aplicações.

Estes pedidos podem ser constituídos por vários sub-pedidos. Por exemplo, formatação de disco; instalação de *software*. A cada sub-pedidos podem estar associados vários acompanhamentos.

Inbox

Quando os utilizadores enviam um *e-mail* para o endereço suporte@fc.ul.pt, é criado um registo de base de dados com os dados do *e-mail* (remetente, destinatário, assunto e mensagem). Existe um estado associado e é possível fazer o acompanhamento das sucessivas respostas. A resolução destes *emails* podem ser atribuídos a um grupo ou a um operador / técnico especifico.

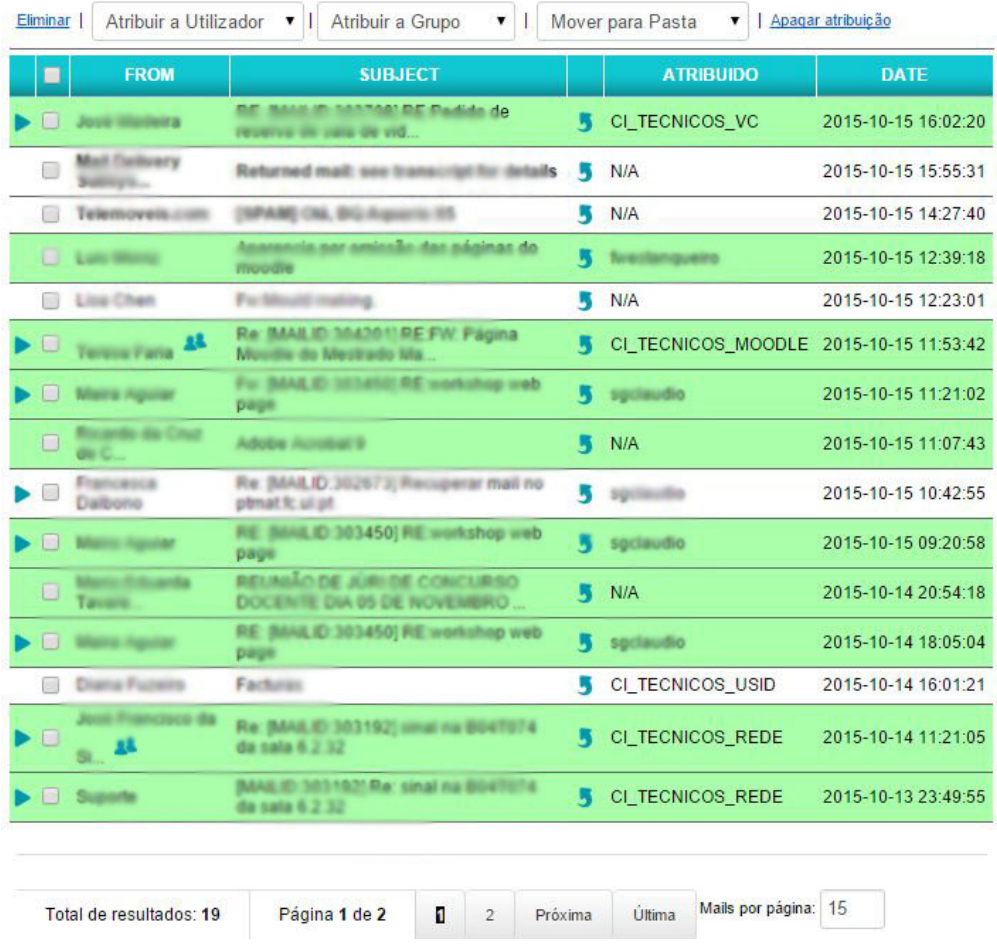


Figura 3 – Back office da aplicação "Inbox"

Gestão de tomadas

Para um docente ou funcionário ligar um computador a uma tomada de rede é necessário configurar o *switch* de modo a indicar o *mac address* da respetiva máquina que está ligada. Para tal, os funcionários podem submeter um pedido. Existem várias possibilidades:

- **Ativação de tomada:** Adiciona o primeiro equipamento a uma tomada livre.
- **Alteração de equipamento:** A tomada já está ativada e acrescenta-se ou altera-se o equipamento que está ligado.
- **Mover Equipamento:** Move equipamento de uma tomada para outra. De notar que esta ação pode originar várias sub-ações como descativar a tomada de origem (fica sem equipamentos) ou cativar a tomada de destino (se estava livre).
- **Remover tomada:** Liberta a tomada de todos os equipamentos.

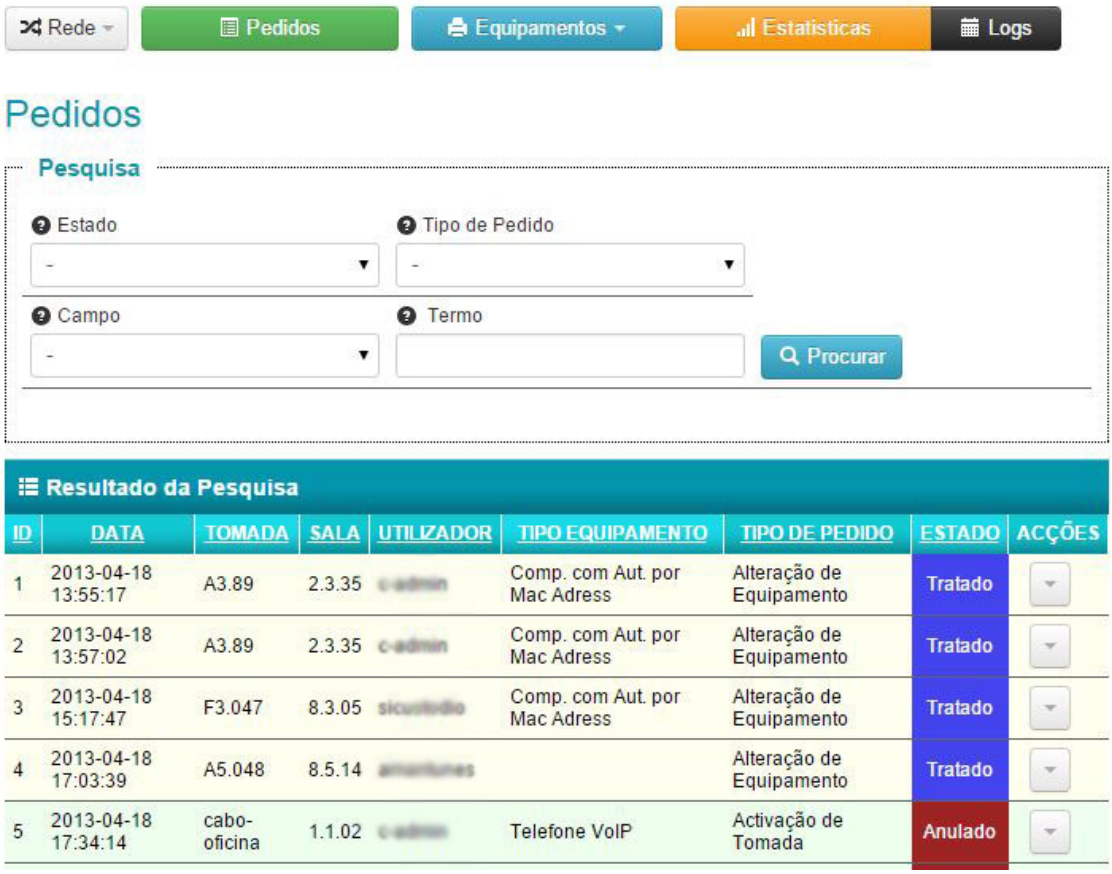


Figura 4 – Back offcie da aplicação de Tomadas

Os operadores de rede tratam dos pedidos em duas fases: Primeiro, determina-se em que *switch* de um determinado bastidor (a aplicação tem uma representação da infraestrutura de rede) à qual a comutação será feita. Segundo, desloca-se fisicamente a esse *switch*, para efetuar a ligação, configurar e fechar o pedido.

Gestão de Pedidos

Permite os responsáveis de departamentos pedir a criação de grupos na AD e gerir os seus membros. Esse grupo poderá ser os Utilizadores de uma determinada impressora. Ao efetuar o pedido, o administrador de sistemas valida ou rejeita a criação desse grupo.

Figura 5 – Back office da aplicação "Gestão de Grupos"

Nova aplicação de Pedidos (GLPI)

GLPI é uma aplicação *web open source* que permite gerir pedidos, inventário do parque informático, etc. Este *software* substituiu as aplicações "Gestão de pedidos" e "Inbox" no início do ano de 2016.

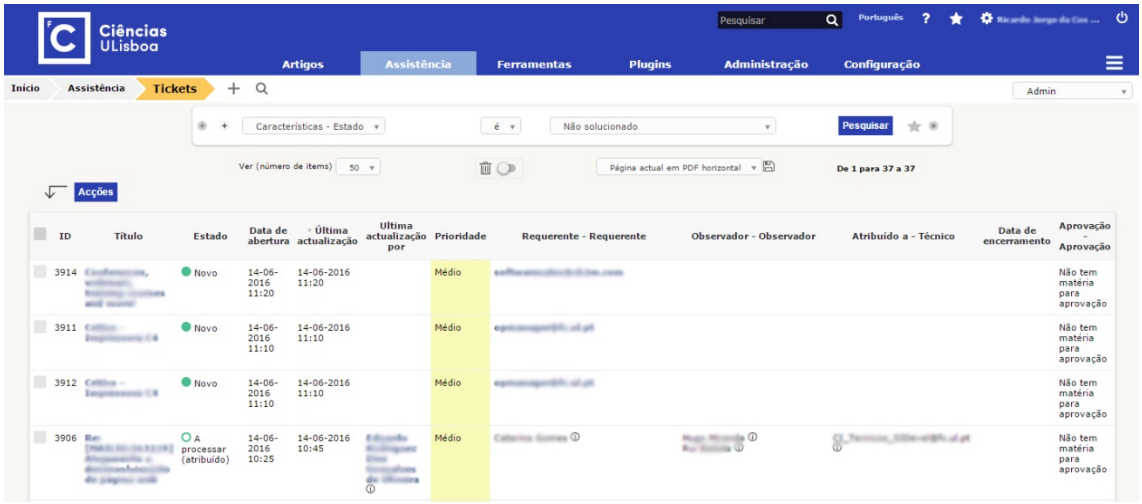


Figura 6 – Back office da nova aplicação de pedidos GLPI

3.2.3 Requisitos

O sistema DW deverá produzir três tipos de relatorios*:*

- **Interno:** Terá a listagem de número de pedidos, tempo gasto e acompanhamentos de todos os departamentos e tempo gasto por cada técnico / operador.
- **Departamental**: Listagem da quantidade de tempo gasto e contagem de pedidos para todos os utilizadores pertencentes a esse departamento / unidade. Também é listado o tempo gasto distribuído por tipo de operação.
- **Público**: Listagem do tempo gasto e contagem de pedidos por operação.

Estes *reports* devem gerados mensalmente e enviados por correio eletrónico.

3.3 Requisitos para o Núcleo de Sistemas de Informação e Desenvolvimento

3.3.1 Funções desempenhadas e necessidades negócio

Este núcleo desenvolve aplicações *ad hoc* para outras unidades funcionais da faculdade e para os funcionários e alunos em geral. Estas aplicações estão implementadas em CakePHP e estão integradas no portal da FCUL que está implementado com o CMS Drupal. Existem aplicações públicas e existem outras que só podem ser acedidas após fazer *login*.

Diariamente são reportados problemas de **utilizadores**, mas muitas vezes é difícil reproduzir todos os seus passos com o intuito perceber a sua causa. É importante existir sistemas de auditoria que registem temporalmente (**data** e **hora)** as **operações** efetuadas bem como o *browser* e sistema operativo utilizado através do ***user agent.***

3.3.2 Origem dos dados

Existem várias proveniências de dados, sendo as mais importantes:

- **SIGES**: Guarda a informação dos alunos.
- **Logos:** guarda a informação dos funcionários.
- ***Active directory*** **(AD):** guarda os utilizadores e os recursos LDAP.
- ***Log*****s** dos servidores aplicacionais: regista as operações efetuadas nos servidores.
- **Base de dados relacionais das aplicações:** todas as aplicações têm uma tabela que registam a ocorrência de pedidos ou *auditing*.

3.3.3 Requisitos

- Centralizar os vários sistemas de auditoria num só em que seja fácil de pesquisar as ações efetuadas por um determinado utilizador no portal *web* da FCUL.
- Necessidade de obter evidências de utilização de determinado utilizador. Para além das operações de escrita, tais como inserções, alterações de registos, etc., que ficam registadas nas bases de dados relacionais, guardar também operações de consulta.

- Ter um registo periódico dos valores e estados das variáveis das aplicações, a fim de analisar as tendências de utilização de determinadas aplicações ao longo do tempo e assim detetar anomalias.
- Perceção de quais os tipos de dispositivos, sistemas operativos e *browsers* através das quais as aplicações são acedidas, a fim de depurar alguns tipos de erros específicos.

3.4 Requisitos para o Núcleo de Infraestrutura de Serviços e Servidores

3.4.1 Funções desempenhadas e necessidades de negócio

Este núcleo realiza a administração de sistemas dos servidores de *email*, *web*, bases de dados, *active direcory*, *proxys*, *domain controllers*, etc da FCUL.

Quando surge a necessidade de procurar evidências de **operações** efetuadas por **utilizadores** num determinado intervalo de **tempo.** Para tal é preciso consultar *logs* desses **servidores** sendo necessário entrar em várias máquinas, utilizar comandos do Unix para tentar procurar o que é pretendido em vários ficheiros. Um outro problema prende-se com a grande parte das entradas dos *logs* não terem qualquer relevância para a despistagem desses problemas, criando bastante ruído e dificultando ainda mais o processo. Seria importante haver uma forma de tratar esses *logs* eliminando os que não têm utilidade para o negócio e centralizá-los num único local melhorando a experiência do técnico a efetuar pesquisas. Estes dados serviriam de histórico de forma a ajudar a aferir os limites máximos de utilização de servidores e justificar, ou não, a aquisição ou o melhor aproveitamento de recursos. Alguns tipos de *logs* como, por exemplo, os dos servidores de e-*mail* ou os *Web* possuem o **número de bytes** da operação. Poderá ser útil contabilizar esses valores a fim de aferir se existe um uso abusivo destes serviços.

Outra informação relevante são as informações relativas ao computador do utilizador (***client machine***), tais como sistema operativo, endereço IP e sua versão (v4 ou v6). Esta informação também ajuda na resolução de erros.

3.4.2 Origem dos dados

A origem dos dados provém dos *logs* dos vários tipos de servidores. Foram só analisados os logs dos servidores web contento uma data e hora, *username*, URL invocado, tipo de operação e *user agent* (ver anexo A - Analise da origem dos dados do OLTP).

3.4.3 Requisitos

- Ter uma interface simples que permita pesquisar as ações efetuadas nos sistemas informáticos da FCUL. Essa pesquisa deve permitir cruzar vários termos como tipo de operação, IP do cliente, intervalo de datas, etc.
- Centralizar a informação dos *logs* num único local.
- Estando os *logs* processados e dentro da DW, podem ser eliminados dos servidores poupando espaço em disco.
- Para certos serviços, guardar o número de bytes gastos na operação com o intuito de detetar usos abusivos da infraestrutura.

3.5 Requisitos para o Núcleo de Infraestrutura de Comunicações

3.5.1 Funções desempenhadas e necessidades de negócio

Este núcleo efetua a manutenção e configuração da infraestrutura da rede física, *wi-fi* e *firewall*. Neste momento existem ferramentas de monitorização de rede que já fazem *reports* e estatísticas em tempo real. Seria conveniente utilizar o DW para fazer *snapshots* destes estados e construir um histórico de monitorização do tráfego de rede utilizado pelas várias unidades e departamentos.

3.5.2 Origem dos dados

A origem dos dados provém das ferramentas de monitorização de rede e da base de dados da aplicação “Gestão de tomadas”.

3.5.3 Requisitos

O núcleo de infraestrutura de comunicações encontra-se num processo de consolidação do uso de ferramentas de monotorização de rede ainda estando numa fase de experimentação e configuração das mesmas. Por esta razão e pelo facto da extração

dos dados por vezes ser complexa, decidiu-se remeter o processo de levantamento de requisitos e posterior desenvolvimento para quando o núcleo achar oportuno.

Capítulo 4 Desenho da arquitetura e escolha dos produtos

Após fazer o levantamento de requisitos, o próximo passo é encontrar a solução tecnologia que melhor sirva as necessidades dos utilizadores. A definição da arquitetura e escolha dos produtos também deverá ter em conta os seguintes fatores [1,5]:

- **Volume de dados:** Vai determinar o número de máquinas necessárias para efetuar o ETL e suportar o DW. Também pode condicionar o tipo de tecnologias.
- **Arquitetura dos sistemas operacionais:** A forma como os dados estão armazenados nos sistemas operacionais condicionam a arquitetura e as decisões técnicas para os obter.
- **Fontes de dados:** A sua localização, tipo e número de fontes de dados tem impacto na arquitetura. Se por exemplo, existir dados externos que têm que ser acedidos, será necessário proceder a implementações para os obter.
- **Orçamento:** O dinheiro disponível determina a escolha de produtos pagos ou *open source* e o número de recursos humanos e de *hardware* alocados ao projeto.

4.1 Escolha de produtos

A premissa mais relevante na construção da *data warehouse* foi o facto de só poder utilizar tecnologias *open source* que não significassem custos para a UI. As aplicações comerciais no mercado são soluções de "chave na mão" com todas as funcionalidades embutidas, como, por exemplo, a Microsoft SSAS e SSIS. O desafio em escolher tecnologias *open source* é arranjar uma combinação de ferramentas que sejam compatíveis entre si e que possam ser integradas facilmente. São necessárias para as seguintes fases:

- **Levantamento de requisitos / análise da origem dos dados**: A fim de compreender melhor os dados de origem, os seus atributos, os seus erros e redundâncias, pode ser útil o auxílio de ferramentas que ajudem a verificar problemas que não são visíveis a olho nu.
- **ETL**: ferramentas de *batch* ou de *data flow* que ajudam na construção e manutenção do ETL. Estas ferramentas têm que permitir o acesso aos tipos de dados de origem existentes e permitir efetuar as transformações de dados necessárias.
- **Cubo OLAP**: ferramenta que permita construir o cubo de dados, permitindo *slice & dice*, *roll-up*, *drill down*, *pivoting*, etc. Esta é talvez das peças mais importantes da *data warehouse* já que é a parte que vai ser acedida pelos utilizadores finais. Tem que ser coerente, escalável, flexível e eficiente.
- **Ferramentas analíticas:** São as aplicações que acedem ao cubo OLAP permitindo a produção de *reports* e efetuar uma análise exploratória dos dados. Estas ferramentas devem ser fáceis de usar e ir de encontro às necessidades dos decisores.
- **Data mining:** São ferramentas que permitem efetuar *data mining* aos dados armazenados na DW, permitindo encontrar informação inesperada que à partida era difícil de encontrar numa análise exploratória normal.

Para cada uma destas fases também existem vantagens e desvantagens que pesam na decisão do que se deve implementar de raiz e o que se deve usar um produto.

Vantagens de usar um produto:

- **Já está implementado e testado**: poupa-se tempo e evita-se erros que possam por em causa a credibilidade da *data warehouse.* Uma solução feita de raiz demorará algum tempo até estabilizar e ficar sem erros críticos.
- **Uniformização e boas práticas**: o uso de um produto que siga as tendências e boas práticas do mercado permite a utilização de paradigmas que são compatíveis com outras *data warehouses*.
- **Desempenho:** Estando estes produtos testados e otimizados, garantem um maior desempenho e eficiência
- **Menor custo de tempo e de mão-de-obra em relação a implementar de raiz:** O custo de configurar e adaptar um produto será bem menor do que o fazer de raiz.

Vantagens de implementar a solução de raiz:

- **Pode-se construir um produto inteiramente à medida dos requisitos:** Não existe qualquer tipo de restrições.
- **Liberdade:** Na escolha das tecnologias a serem usadas, linguagem de programação, etc.
- **Permite uma melhor manutenção e crescimento:** Tendo sido o levantamento de requisitos bem efetuado e adaptado às necessidades da organização, o crescimento da DW será facilitado.
- **Os produtos podem estar mal documentados e serem difíceis de configurar:** Apesar de se poupar tempo porque a ferramenta já está implementada, pode-se perder bastante tempo a alterar os ficheiros de configuração para por o produto a funcionar. Essa dificuldade, ocorre sobretudo em ferramentas *open source* em que a documentação é escassa. Poderá ser complexo configurar especificidades inerentes a regras de negócio e ou restrições tecnológicas.
- **Produto fechado e licenciamento:** Para implementar regras de negócio muito específicas, pode ser difícil ou até mesmo impossível por ser um programa proprietário em que não se tem o código fonte. Mesmo sendo *open source* o licenciamento pode ter restrições referentes à modificação do código.

Tendo estes fatores em conta fez-se um estudo das ferramentas *open source* existentes no mercado (lista ordenada com os *links* para os respetivos *websites* no Anexo B – Listagem de ferramentas para BI) :

Plataformas e soluções integradas:

Produto	**Observações**
BEE / gooddata	Plataforma BI em cloud
Pentaho	Soluções integradas BI
Hadoop	Solução para processamento distribuído de *big data*
Hive	Ferramenta de *big data* para SQL
Hortonworks	Solução integrada para Hadoop

Tabela 1- Plataformas BI e soluções integradas

Cubos OLAP:

Produto	Observações
Mondrian	Cubo OLAP baseado na ferramenta comercial Microsoft SSAS e apadrinhado pela Pentaho
Kylin	Cubo OLAP para Hadoop
Olap4j	Cubo OLAP para JAVA
PhpMyOlap	Cubo OLAP para PHP.

Tabela 2 - Cubos OLAP

Data Integration e ETL:

Produto	Observações
Pentaho Kettle	Ferramenta de *data integration* da Pentaho
Jaspersoft ETL	Ferramenta de ETL
KETL	Ferramenta ETL baseada em XML

Tabela 3 - Data Integrations e ETL

Ferramentas analíticas e reporting:

Produto	Observações
opernI	Ferramenta analítica orientada para *big data*
BIRT	Ferramenta de *reporting* para BI que é um *plugin* Eclipse
jasperSoft	*Reporting* e *business analytics*
jpivot	Aplicação web de *pivot tables* que integra com Mondrian
Claudera	Ferramenta de acesso a dados provenientes do Hadoop
Saiku	Ferramenta analítica para a web que comunica com JDBC ou com XMLA

Tabela 4 - Ferramentas analíticas e de reporting

Data Mining

Produto	Observações
R	Linguagem de programação para tratamento de dados que incluiu algumas funcionalidades de *data mining*
Weka	Coleção de algoritmos de *data mining* que pode ser integrado com JAVA

Tabela 5 - Data mining

A análise efetuada permite perceber que existem muitas soluções *open source* no mercado e que podem ser agrupadas em três grandes famílias: Hadoop, Pentaho e transversais.

Ponderou-se o *Hadoop* e a sua família de produtos pelo facto de poder ajudar na questão de tratar muitos *logs* dispersos por vários servidores, mas foi posto de parte porque a informação gerada pelos sistemas operacionais não justifica dado o seu volume diminuto. O *Hadoop* é uma boa ferramenta para *big data*, mas é mais difícil de implementar. A família Pentaho possui ferramentas que são pagas e outras que são gratuitas. As gratuitas são mais limitadas e têm uma documentação muito reduzida. Relativamente às ferramentas transversais, têm a vantagem que poderem ser integradas com qualquer outra existindo produtos com maior e menor qualidade.

4.1.1 Ferramentas para o levantamento de requisitos / análise da origem dos dados

Sendo a origem dos dados bases de dados relacionais e ficheiros de *log*, foi utilizado o cliente *Navicat* para aceder às bases de dados. Para estudar os *logs*, utilizou-se os comandos *awk* e *grep* e ainda pequenos programas JAVA para fazer *parsing* dos *logs* e inseri-los numa base de dados relacional.

4.1.2 ETL

O ETL tem que permitir efetuar a extração e transformação necessárias dos dados e deve ser um processo que possa correr periodicamente e de forma incremental. O **Pentaho Kettle** cumpre esses requisitos. Como permite invocar *web services*, caso existam operações de maior complexidade, estas podem ser remetidas para um desenvolvimento mais clássico implementando serviços e encapsulando-os.

4.1.3 Cubo OLAP

Foi escolhida como Cubo OLAP o **Mondrian**. O Mondrian é a versão gratuita da *Pentaho*. Foi implementada em JAVA e integra com muitas origens de dados através do JDBC. É possível definir cubos de dados através de um *schema* em formato XML. Possui nativamente a extensão de SQL *Multidimensional Expression* (MDX) criado pela Microsoft. Possui ainda o *standard* XMLA que permite a conectividade com muitas outras ferramentas analíticas e até o Excel. As vantagens são:

- **Tecnologia *open source*:** não acarreta custos e permite interagir com outras tecnologias gratuitas.

• **Baseada na ferramenta comercial MS SSAS:** Os utilizadores que usaram esta ferramenta tem facilidade em usar o *mondrian* e vice-versa.

• ***Schema* Lógico em XML**: permite definir vários cubos de dados, abstraindo-se das origens de dados. Permite também criar *labels* mais inteligíveis dos campos, bem como criar medidas virtuais com outros tipos de agregação e formatação.

• **Permite ainda criar cubos virtuais que juntam vários outros cubos** permitindo consultar vários *Data Marts* integrados num só.

• **MDX**: é um *standard* que estende o SQL a fim de lidar com *queries* analíticas de uma forma mais simples e transparente. A mesma *query* em MDX tem muito melhor desempenho que a equivalente em SQL normal por estar optimizada para fazer agregações.

• **XMLA**: interoperabilidade com outras ferramentas (por exemplo: Saiku, Pentaho, etc) compatíveis com esta tecnologia. A comunicação é feita com SOAP ao invés de se ligar diretamente à base de dados.

• **Suporte a olap4j:** *Standard* OLAP que permite a conectividade com soluções de mercado baseadas em JAVA.

• **Desempenho:** O acesso aos dados é rápido comparativamente a outros produtos *open source* existentes.

4.1.4 Ferramentas analíticas

O cubo de dados deverá ser suficientemente flexível para permitir o maior número de ferramentas possíveis, permitindo aos utilizadores finais usarem as suas ferramentas prediletas. A escolha do Mondrian como Cubo OLAP e a modelação dimensional permite usar as seguintes aplicações:

- **Excel:** Através de *Power Pivot* ou *Pivot Table*, selecionando uma fonte de dados compatível com XMLA.
- **Jpivot:** *Servlet* que vem com o *Mondrian*. Apesar de ser uma aplicação web simples é poderosa e fácil de usar.
- **Saiku:** aplicação *web* com *design* elegante que consegue integrar com o *mondiran* através de XMLA.
- **Relatórios em CakePHP e DomPDF:** apesar de haver ferramentas especializadas em *reporting*, dada a especificidade dos *layouts* a serem gerados (logotipo da faculdade e estrutura do documento) e a necessidade

de criar um processo que os envie automaticamente por *email* em formato PDF, decidiu-se enveredar por uma solução conhecida e já implementada na UI. O cakePHP é uma *framework* MVC para PHP e o DomPDF, é uma implementação do Google que transforma HTML em PDF.

4.1.5 Data mining

Optou-se por usar o **weka**, por ser uma ferramenta com uma boa variedade de algoritmos de classificação e *clustering* e por ser fácil de criar os ficheiros de *input* (em formato arff).

O weka é um pacote de *software* implementado em JAVA que possui vários algoritmos aplicáveis no *data mining* e *machine learning*.

4.2 Arquitetura

Existem quatro tipos de arquitetura possíveis [5]:

- **Arquitetura a um nível:** Numa única máquina é suportada os sistemas operacionais e todas as componentes da DW. Solução mais económica, mas com sérias limitações de desempenho e de recursos.
- **Arquitetura a dois níveis:** Existe uma máquina dedicada a todos os componentes referentes à DW. Os sistemas operacionais localizam-se noutra (s) máquinas. Esta solução continua a ser económica e tem menos limitações que a solução a um nível porque não interfere com o desempenho dos sistemas operacionais quando por exemplo o processo de ETL está a ser executado.
- **Arquitetura a três níveis**: Para além dos sistemas operacionais existe uma máquina dedicada ao ETL e processamento de dados distinta da máquina de apresentação dos dados. Esta solução é mais dispendiosa, mas garante uma alta disponibilidade de acesso aos dados da DW.
- **Arquitetura a quatro níveis**: Semelhante à de nível três, mas com uma camada adicional de *middleware* que permite que os dados da DW sejam acedidos fora da instituição (via *web services* por exemplo).

Deliberou-se adotar uma arquitetura a dois níveis com uma máquina dedicada exclusivamente às atividades da DW. Eventualmente com o crescimento e amadurecimento desta implementação, se enverede por uma arquitetura a três níveis ou

até mesmo de quatro. Dadas as escolhas tecnológicas e as restrições *open source*, esta máquina dedicada deverá ter os seguintes requisitos:

- **Servidor Linux:** Preferencialmente da família Red hat ou Debian.
- **Ferramenta "yum" ou "apt-get" instalada no servidor:** Para instalar de uma forma simples os outros requisitos.
- **Duas Bases de dados de Mysql:** uma para a *data staging area* e outra para a *data presentation area.*
- **Acesso a todas as origens de dados:** O servidor deverá conseguir aceder a todas as origens de dados dos sistemas operacionais, implicando alterações de configurações de Firewall, *roles* de servidores, etc.
- **JRE atualizado:** O saiku, Mondrian e Jpivot estão implementados em JAVA e necessitam da versão mais recente.
- **Apache** e **Apache Tomcat:** O Mondrian e o JPivot funcionam num servidor aplicacional JAVA.
- **JDBC para o MySQL**: O Mondrian precisa do driver de MySQL do JAVA para comunicar com as BD's.
- **Pentaho Kettle instalado no servidor:** Embora se possa construir remotamente o ETL numa outra máquina, para o Kettle ser executado tem que estar instalado no servidor.
- **Mondrian e Jpivot:** A distribuição do Mondiran com o JPivot embutido deverá estar instalada no Apache Tomcat.
- **Saiku (Opcional):** O saiku pode correr localmente na máquina do cliente (criando uma instância de Tomcat), mas casos se pretenda um endereço permanente do lado do servidor este deverá ser instalado.

Capítulo 5 Processos de negócio

Após ter efetuado uma análise à Unidade de informática, é preciso identificar quais os processos de negócio aos quais se pode construir um sistema OLAP, definir as perguntas analíticas das quais esse sistema deverá conseguir responder e escolher o processo prioritário [1].

5.1 Identificação dos processos de negócio

Foram identificados quatro processos de negócio cada um associado ao seu respetivo núcleo da UI.

5.1.1 Pedidos

Associado à área do Núcleo de Suporte a Utilizadores, E-learning e Multimédia e tem como objetivos principais compreender quem são os utilizadores que fazem pedidos, determinar o tempo gasto dos técnicos e operadores e gerar *reports* mensais a fim de mostrar evidencias do trabalho efetuado pela UI.

5.1.2 Aplicações

Está diretamente associada com o núcleo de Sistemas de Informação e Desenvolvimento e tem como objetivos principais registar as operações efetuadas nas diversas aplicações criando um sistema de *auditing* e conhecer os tipos de dispositivos, sistemas operativos e *browsers* que acedem às aplicações.

5.1.3 Sistemas

Está diretamente associada com o Núcleo de Infraestrutura de Serviços e Servidores e tem como objetivos principais criar um sistema em centralize a pesquisa por *logs* de servidores descartando as entradas dos *logs* que já não tem utilidade libertando espaço em disco nos servidores. Tem também como objetivo contabilizar o valor dos bytes de certas operações no caso dos *logs* devolverem essa informação.

5.1.4 Redes

Está diretamente associada com o Núcleo de Infraestrutura de Comunicações e tem como principais objetivos guardar um histórico dos dados produzidos pelas ferramentas de monotorização de rede.

5.2 Perguntas analíticas

Antes de proceder à escolha do sistema prioritário, são definidas algumas preguntas analíticas ao qual o sistema OLAP deverá conseguir responder:

#	**Pergunta**
1	Qual o tipo de operações que despendem mais tempo?
2	Quais são as operações mais frequentes?
3	Qual a distribuição de operações ao longo de um ano letivo / semestre?
4	Qual a distribuição do tempo gasto em relação aos departamentos?
5	Existem muitos pedidos pendentes?
6	Quem tipo de utilizador despende mais recursos? Alunos, funcionários ou docentes?
7	Qual o canal que consome mais recursos?
8	Existe alguma correlação entre os recursos despendidos e a faixa etária dos utilizadores?
9	Existe alguma correlação entre o tempo gasto e o género?
10	Existe uma correlação entre o número de acompanhamentos e o tempo gasto?
11	Quais os tipos de operação com mais acompanhamentos?
12	Quais as nacionalidades dos utilizadores não portugueses mais comuns e os recursos que consomem?
13	Qual a relação entre o número de pedidos de utilizadores externos e internos?
14	Existe alguma correlação entre um tipo de operação e um canal preferencial?
15	Qual é efetivamente o uso das aplicações nos fins-de-semana e feriados?
16	Qual o período do dia com mais atividade? E com menos?
17	Existem diferenças de tempo gasto entre técnicos e operadores?
18	Qual é o tipo de dispositivo, sistema e *browsers* mais comuns a aceder às aplicações da FCUL?
19	Quais as operações de consulta e de escritas mais acedidas? E as menos?
20	Quais são as operações de consulta e de escrita que são mais repetitivas?
21	Quais as aplicações que têm picos de utilização ao longo do ano letivo?

22	Quem usa mais o portal da FCUL? Alunos, docentes, funcionários ou anónimos?

Tabela 6 - Perguntas analíticas

Relevância das perguntas analíticas para cada processo de negócio:

Pergunta	Pedidos	Aplicações	Sistemas	Redes
1	X			
2	X	X	X	
3	X	X	X	
4	X			
5	X			
6	X			
7	X			
8	X			
9	X			
10	X			
11	X			
12	X	X		
13	X			
14	X	X		
15	X	X		
16	X	X		
17	X	X		
18		X		
19		X	X	
20		X	X	
21		X	X	
22		X	X	

Tabela 7- Relevância das perguntas analíticas para cada processo de negócio

5.3 Escolha do processo prioritário

O diagrama de exequibilidade *versus* importância ajuda a determinar a ordem pela qual os processos devem ser implementados:

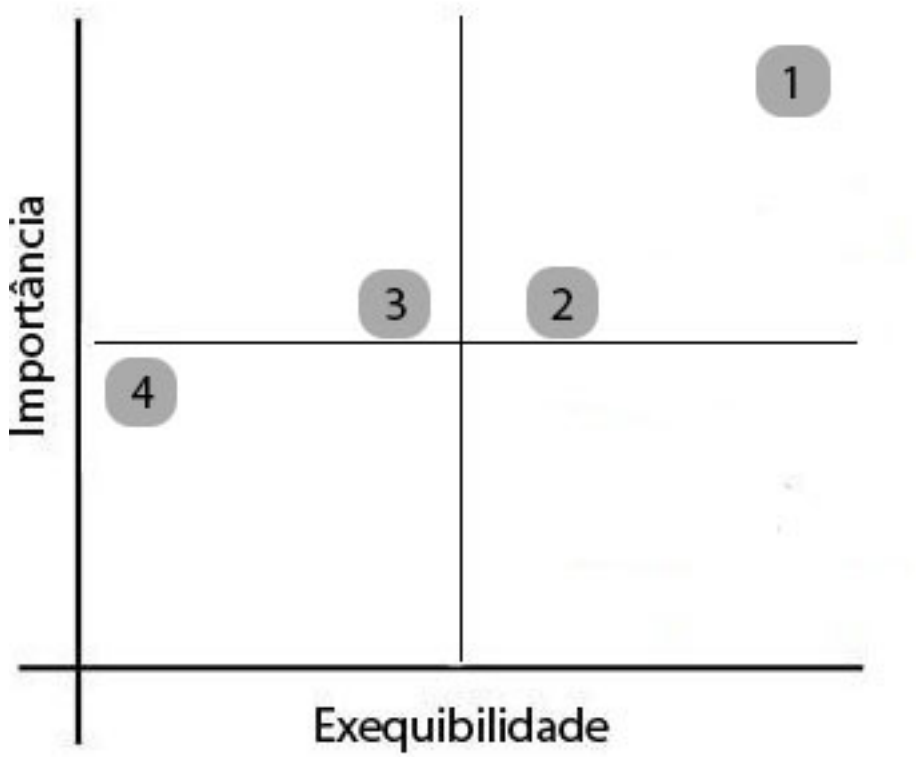


Figura 7- Matriz exequibilidade versus Importância

1. **Pedidos:** É o processo de negócio no qual é mais prioritário extrair informação. É o mais simples de implementar porque as origens de dados são bases de dados relacionais bem conhecidas e utilizadas no dia-a-dia.

2. **Aplicações:** É um processo importante, mas tem uma exequibilidade mais baixa porque implica cruzar informação de bases de dados relacionais com *logs*. Note-se que os *logs* têm o mesmo formato por serem servidores *web* idênticos e estão centralizados num só local.
3. **Sistemas:** Apesar de ter tanta importância como o processo das aplicações, é de implementação mais complicada, porque existe mais variedade de *logs* e estão mais dispersos.
4. **Redes:** De difícil implementação porque as ferramentas de auditoria de rede são variadas e muitas delas proprietárias, sendo complicado obter informação.

O processo de negócio “Pedidos” será o primeiro a ser implementado. Os processos de negócio “Aplicações” e “Sistemas” vão ser implementados assim que se terminar o levantamento exaustivo dos tipos de *logs* existentes e a sua localização. O processo de negócio “Redes” será implementado assim que o uso das ferramentas de monitorização de rede estiverem consolidadas e se conseguir extrair os dados.

Para o contexto deste PEI decidiu-se documentar todas tarefas efetuadas. Por essa razão, o ciclo de vida do processo de negócio “Pedidos” é totalmente documentado, nos processos “Aplicações” e “Sistemas” é documentado o levantamento de requisitos e modelação dimensional. O processo de negócio “Redes” é mencionado, mas não será efetuada qualquer ação.

Capítulo 6 - Modelação dimensional

6.1 Enquadramento teórico

A modelação dimensional [1,4] é uma técnica amplamente usada para apresentar os dados analíticos. Um dos objetivos principais é construir um modelo de dados simples e intuitivo para os decisores. Ao contrário dos sistemas operacionais que tendem a ter modelos de bases de dados relacionais normalizados (3NF), em que as tabelas estão organizadas de maneira a haver uma menor redundância de dados possível, originando o *snowflaking,* a modelação dimensional tende a utilizar regras totalmente opostas. Neste caso é proposto um diagrama em estrela em que várias tabelas (dimensões) se ligam a uma única (a tabela de factos). Mais uma vez o intuito é tornar o sistema de informação mais intuitivo e como existem menos junções entre tabelas, existem um melhor desempenho.

6.1.1 Dimensões

As dimensões são tabelas que representam entidades relevantes para o negócio, como utilizador, produto, etc. Essas dimensões vão possuir vários atributos relevantes ao contexto do *data mart*. Exemplo: para a dimensão Utilizador: nome, departamento, etc. Estes atributos também podem funcionar como filtros em *queries* analíticas. Estas dimensões podem ter chaves naturais que representam univocamente cada elemento. Por exemplo, o número de bilhete de identidade ou número de funcionário. No entanto, é aconselhado criar uma nova chave substituta (*surrugate key*) que represente o elemento no DW de forma independente quer às chaves naturais, quer sobretudo às chaves que podiam existir no sistema operacional. Isto permitirá uma maior independência quer caso o sistema cresça e existam vários *data marts*, quer no caso em que ocorram alterações no sistema operacional. Uma desvantagem é tornar bastante mais complexo o processo de ETL, mas mesmo aqui também se encontra a vantagem de este ficar mais independente dos sistemas operacionais.

6.1.2 Factos

A tabela de factos une-se com as várias dimensões através de chaves estrangeiras e possui ainda medidas que guardam valores relevantes a cada transação. Se as dimensões

se podem comparar com substantivos, a tabela de factos pode representar um verbo ou uma ação: É importante especificar a granularidade da tabela de factos. Uma granularidade pequena gera muita informação, mas pode-se ir a um maior detalhe. Quanto mais pequena for a granularidade melhor, se bem que a partir de certo nível de detalhe esta deixa de ter utilidade para os decisores e prejudica a análise dos dados. Associado a essas ações existem medidas: um valor de uma venda, quantidade, volume de Bytes, etc. Estas medidas devem ser aditivas para poderem ser agregáveis: Por exemplo, o valor de vendas pode-se somar, o valor de temperaturas não se pode agregar. Existem ainda medidas semi-aditivas que mediante certas circunstâncias podem-se agregar. Existem três tipos de tabelas de factos:

- **Transacionais:** Registam eventos que ocorreram. Por exemplo, X comprou Y na data Z com preço W.
- **Instantâneos periódicos:** Guardam dados acumulados em períodos fixos e regulares. Por exemplo, vendas de Março
- **Instantâneos cumulativos:** Acompanham um processo recorrente, mas de duração variável. Por exemplo, ciclo de vida de um produto.

Dimensões conformadas

Caso a *data warehouse* cresça, podem existir vários *data marts* que usem a mesma dimensão. Nos *data marts* pode ser relevante um determinado conjunto de atributos, ao passo que noutros *data marts* possa ser outro conjunto. Pode haver ainda casos que existam conflitos entre atributos, ou seja atributos que têm o mesmo nome, mas que significam coisas diferentes.

Para mitigar este problema é importante conformar as dimensões, colocando todos os atributos revelantes para todos os *data marts* e eliminando os conflitos. Para tal existirá uma entidade denominada “Gestão de dados mestres” que tem como objetivo dialogar com os vários *data marts* a fim de decidir os campos que interessam através de matrizes de processos e relatórios transdepartamentais.

Estas dimensões conformadas podem gerar dimensões muito grandes. Para além de tirarem eficiência, dificultam a sua legibilidade. Existem duas opções possíveis:

- **Bifurcações:** faz-se algum *snowflaking* e divide-se a dimensão numa dimensão principal que liga a outras sub-dimensões. Por exemplo: na dimensão utilizador, para um *data mart* interessa a morada de faturação e para outro a morada de entrega de encomenda. Neste caso, as moradas são retiradas da tabela que representa a dimensão utilizador e são criadas duas novas tabelas de morada que ligam à tabela da dimensão utilizador

- **Mini Dimensões:** A dimensão é partida em várias sub-dimensões que ligam diretamente à tabela de factos.

As dimensões conformadas têm como objetivo final permitir o *drill-across*. Ao passo que no cubo de dados o *drill down* e *o roll-up* permite navegar verticalmente na DW, o *drill-across* permite navegar horizontalmente entre os vários *data marts*.

6.1.3 Role-playing

A mesma dimensão pode ser aplicada em significados diferentes. Por exemplo, data de início e data de fim. Nestes casos não faz sentido ter duas dimensões que têm exatamente a mesma informação. Por esta razão, esta dimensão será a mesma tabela e recorre-se a *views* ou *alias* para dar a ilusão que existe várias. Na tabela de factos haverá uma chave primária para a dimensão original e outra para a *view* / *alias* criado.

6.1.4 Dimensões de mudanças lenta

Ao longo do tempo os valores dessas dimensões podem mudar. É importante atualizar a informação sem comprometer a informação anterior por questões de histórico. Para resolver estas questões existem três formas [1,4,5]:

- **Tipo 1:** Substituir diretamente o valor na tabela de dimensão. Útil para corrigir erros detetados. Por exemplo, o número telefone incorreto, ou para atributos artificias que dependem de outros como a idade (que depende da data de nascimento). A vantagem é ser de simples implementação e a desvantagem é não guardar histórico.
- **Tipo 2**: Acrescentar uma linha na tabela de dimensão. Mantêm-se a linha antiga, e cria-se uma nova com as alterações. Por exemplo, a alteração de morada cliente. É preciso criar colunas adicionais para determinar que linha está ativa e em que período as outras linhas estiveram. A vantagem é guardar histórico e a desvantagem é ter uma implementação mais complicada.
- **Tipo 3**: acrescentar colunas à tabela de dimensão. Por exemplo, criar uma coluna chamada “morada nova”. Tem a desvantagem de tirar legibilidade à dimensão. Pode ser útil no caso concreto em que seja importante comparar lado a lado o campo antigo e o campo novo.

6.2 Data mart pedidos

Aplicando a modelação dimensional para o *data mart* dos pedidos temos das dimensões data, relógio, utilizador, operação, canal, estado e a tabela de factos conforme o diagrama abaixo:

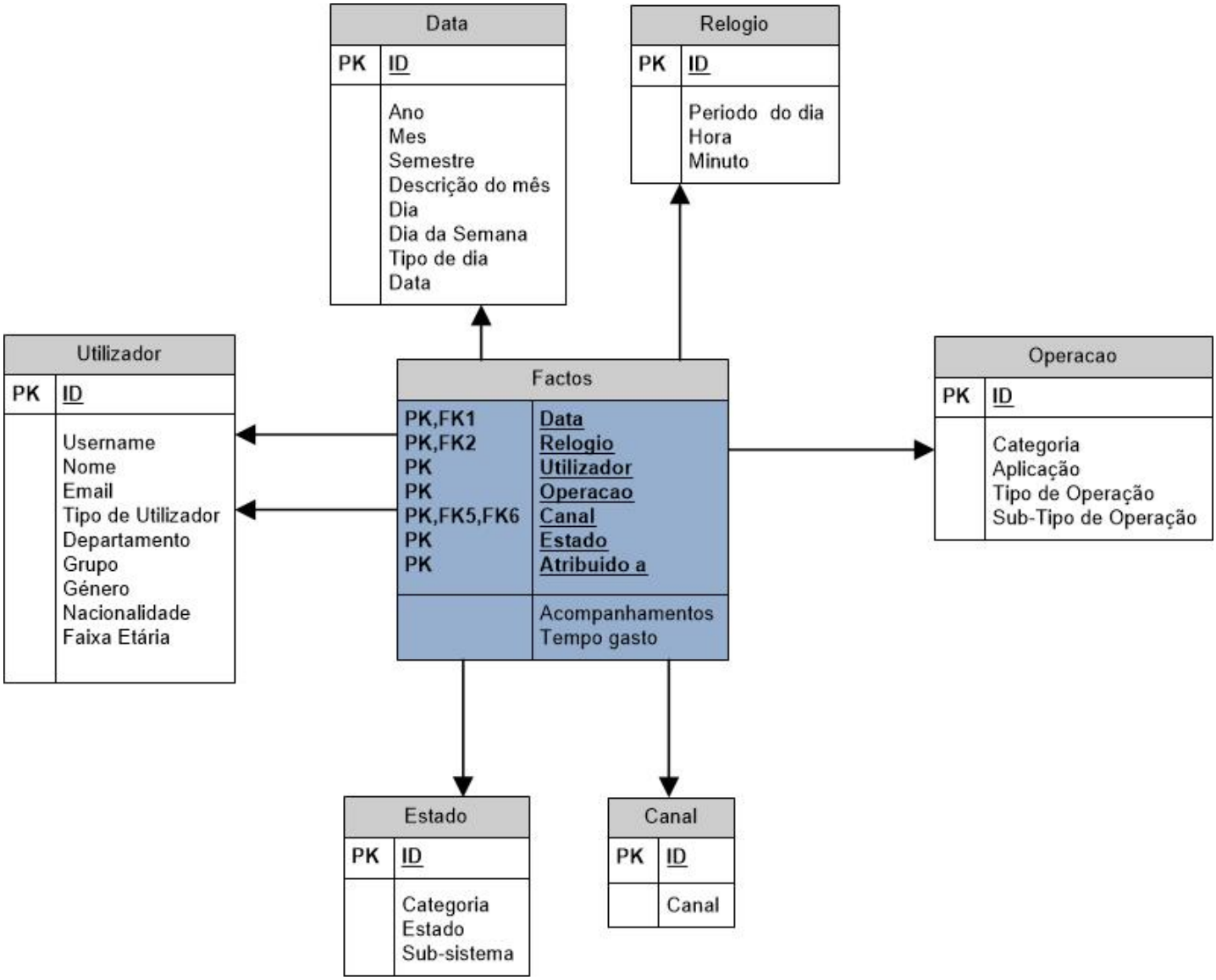


Figura 8- Star schema do data mart pedidos

6.2.1 Dimensão data

Sendo uma das dimensões mais importantes de um DW, a dimensão data é constituída pelos seguintes campos:

Nome	Tipo	Exemplo
Ano	INTEIRO	2015
Semestre	TEXTO	1º Semestre
Mês	INTEIRO	1
Descrição do mês	TEXTO	Janeiro
Dia	INTEIRO	1
Dia da Semana	TEXTO	5ª Feira
Tipo de dia	TEXTO	Feriado
Data	DATA	01/01/2015

Tabela 8 - Atributos da dimensão data

O semestre é obtido através da BD académica SIGES e é importante no contexto de uma Universidade.

O tipo de dia é obtido na aplicação "marcação de férias " que assegura para além dos feriados, as tolerâncias de ponto que foram dadas.

O campo data existe meramente para permitir ao utilizador filtrar por uma data específica. Sendo mais fácil de indicar a data ="01-01-2015" do que Ano ="2015", Mês = "01" e Dia= 1"

Hierarquias:

Nome	Hierarquias
Letivo	**ANO => SEMESTRE => MÊS => DIA**
Legal	**ANO => MÊS => DIA**
Tipo	**ANO => SEMESTRE => MÊS => TIPO DE DIA => DIA DA SEMANA**

Tabela 9 - Hierarquias da dimensão data

6.2.2 Dimensão relógio

Esta dimensão foi criada à parte da Dimensão Data por duas razões:

1. **Economia de recursos:** Se fosse só uma dimensão existiria 1440 entradas na BD por cada dia existente. Assim estas 1440 entradas (uma por cada minuto do dia) são constantes ao passo que a Dimensão Data vai crescendo ao longo dos tempos
2. **Permite mais opções nas *queries* analíticas:** Por exemplo, saber a variação de utilização de determinada aplicação ao longo do dia para o ano de 2015.

Nome	Tipo	Exemplo
Período do dia	TEXTO	Tarde
Hora	INTEIRO	18
Minuto	INTEIRO	25

Tabela 10 - Atributos da dimensão relógio

O período do dia permite agrupar várias horas em conjuntos:

- **Manhã**: entre as 6 e as 13 horas
- **Tarde**: entre as 13 e as 20 horas
- **Noite**: entre as 20 e as 6 horas

Hierarquias:

Nome	Hierarquias
	PERIODO => HORA=> MINUTO

Tabela 11- Hierarquias da dimensão relógio

6.2.3 Dimensão utilizador

Esta dimensão guarda a informação referente ao utilizador.

Nome	Tipo	Exemplo
Username	TEXTO	rjsimoes
Nome	TEXTO	Ricardo Jorge da Costa Simões
Email	TEXTO	rjsimoes@fc.ul.pt
Tipo de Utilizador	TEXTO	FUNCIONARIO
Departamento	TEXTO	Unidade de Informática
Grupo	TEXTO	Técnicos

Género	TEXTO	Masculino
Nacionalidade	TEXTO	Portugal
Faixa Etária	TEXTO	[31-40]

Tabela 12 - Atributos da dimensão utilizador

O atributo “Username” é uma chave natural que representa o Utilizador no sistema. A finalidade do campo “Email” é ter um meio de contacto caso seja necessário.

O “Tipo de utilizador” pode ser: “FUNCIONARIO”, “DOCENTE“, ”ALUNO”, ”EXTERNO”.

O atributo “departamento” é preenchido dependendo se o utilizador for:

- **Funcionário:** o departamento ou unidade onde trabalha.
- **Docente:** o departamento onde leciona.
- **Aluno:** o departamento do curso onde está matriculado.
- Caso seja externo ou noutra situação em que não seja possível aferir qual o departamento é preenchido “Desconhecido”
- O grupo pode ser : “Operador”, “Técnico” ou “Nenhum”.

As faixas etárias são:

- [18-20], [21-23], [24-26], [27-30], [31-40], [41-50], [51 – 65], [65+]

Assume-se que um aluno universitário tem mais de 18 anos. Mesmo que aconteça casos excecionais são colocados no intervalo [18-20].

Os intervalos têm amplitudes mais pequenas nas idades mais baixas e vão alargando devido ao facto de por exemplo um aluno caloiro ser completamente diferente de um aluno finalista. Nas idades mais avançadas essas diferenças tornam-se mais irrelevantes.

Hierarquias:

Nome	**Hierarquias**
Dados utilizador	**NACIONALIDADE => GÉNERO=> FAIXA ETÁRIA => NOME**
Departamento	**DEPARTAMENTO => GRUPO => NOME**
Faixa etária	**FAIXA ETÁRIA => NOME**
Género	**GÉNERO => NOME**
Tipo de Utilizador	**TIPO DE UTILIZADOR => NOME**

Tabela 13 - Hierarquias da dimensão Utilizador

6.2.4 Dimensão operação

Esta operação descreve a operação que foi realizada:

Nome	Tipo	Exemplo
Categoria	TEXTO	HARDWARE
Aplicação	TEXTO	Gestão Pedidos
Tipo de Operação	TEXTO	Hardware
Sub-Tipo de Operação	TEXTO	Substituição de disco

Tabela 14 - Atributos da dimensão operação

Caso o "subtipo " não seja usado é preenchido como "Não Aplicável"

Hierarquias:

Nome	Hierarquias
Por Aplicação	**APLICAÇÃO => TIPO DE OPERAÇÃO => SUB-TIPO DE OPERAÇÃO**
Por Tipo	**CATEGORIA => TIPO DE OPERAÇÃO => SUB-TIPO DE OPERAÇÃO**

Tabela 15 - Hierarquias da dimensão operação

6.2.5 Dimensão estado

Descreve o estado do pedido

Nome	Tipo	Exemplo
Categoria	TEXTO	Concluído
Estado	TEXTO	Fechado
Sub-Sistema	TEXTO	Pedidos

Tabela 16 - Atributos da dimensão estado

O campo "Estado" é exatamente o mesmo que vem da aplicação. O campo "Categoria" é a generalização lógica do estado. Manteve-se os dois para por um lado não haver ambiguidades, mas ao mesmo tempo também poder generalizar os estados. O campo "sub-sistema" indica a que aplicação pertence o estado.

Hierarquias:

Nome	Hierarquias
	ESTADO GENÉRICO => ESTADO

Tabela 17- Hierarquias da dimensão estado

6.2.6 Dimensão canal

Descreve o estado canal em que o pedido foi efetuado: WEB, Correio eletrónico, Presencial ou telefone.

Nome	Tipo	Exemplo
Canal	TEXTO	Web

Tabela 18- Atributos da dimensão canal

Hierarquias:

Nome	Hierarquias
	CANAL

Tabela 19- Hierarquias da dimensão canal

6.2.7 Dimensão operador

Existe *role-playing* relativamente a esta dimensão. Este *alias* é criado a partir de uma *view* chamada "dim_operador" e com a *quer*y de SQL: "*select * from dim_utilizador where grupo <> 'Nenhum' "*.

6.2.8 Factos:

A tabela de factos tem como granularidade uma operação: O utilizador U realizou a operação O através do canal C na data D, hora R e com estado E.

A tabela de factos tem as seguintes medidas:

- **Tempo gasto:** Tempo gasto em minutos.
- **Acompanhamentos:** número de acompanhamentos por pedidos.
- **Contagem:** Esta é uma medida virtual criada no cubo de dados e utiliza o *count* como forma de agregação.

6.3 Data mart aplicações

Aplicando a modelação dimensional para o *data mart* dos pedidos temos das dimensões data, relógio, utilizador, operação, canal e estado e a tabela de factos conforme o diagrama abaixo:

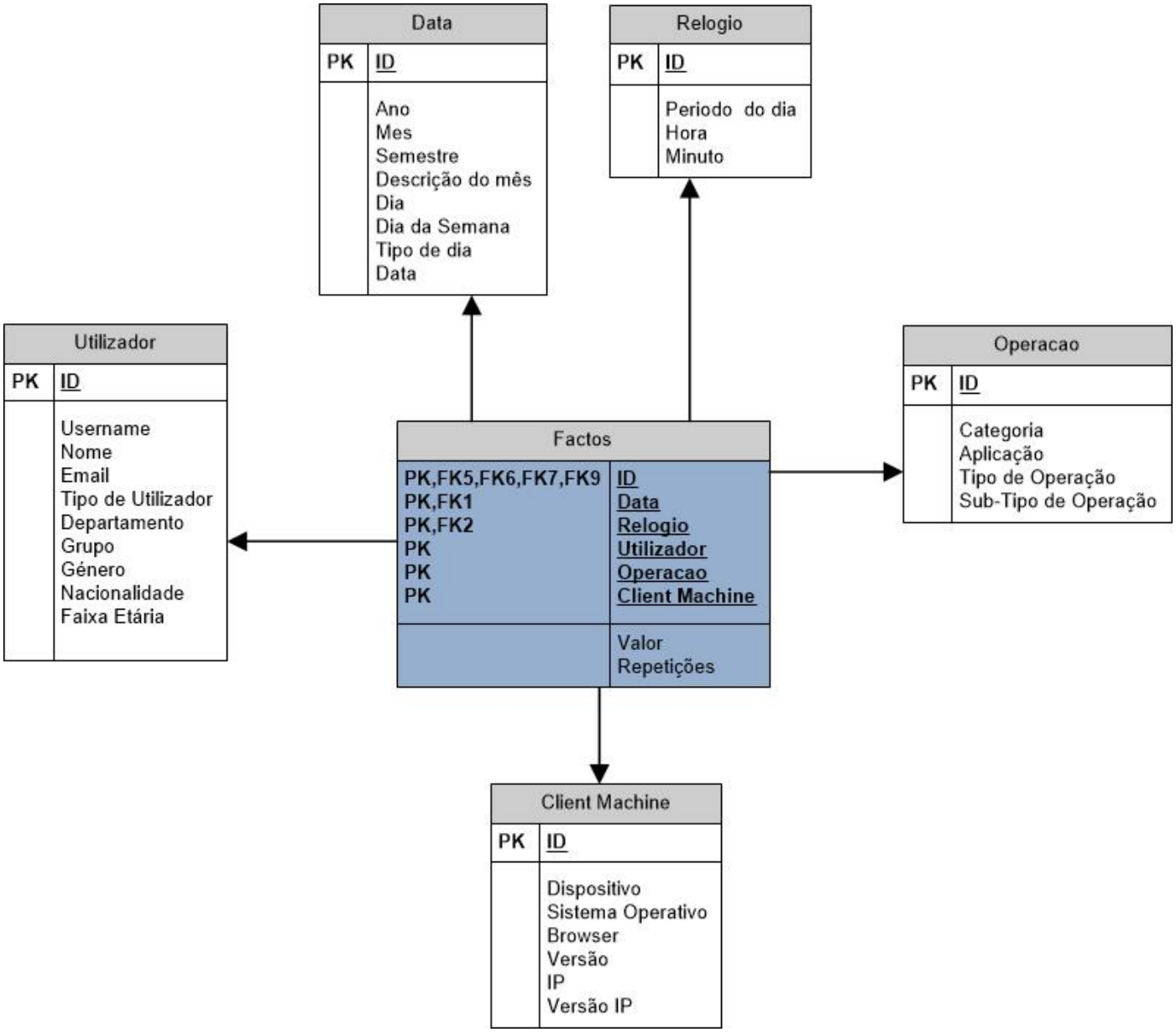


Figura 9 - Star Schema do Data Mart aplicações

6.3.1 Dimensões data, relógio, utilizador e operação

As dimensões Data, Relógio, Utilizador e Operação estão conformadas com as equivalentes usadas no *data mart* Pedidos.

6.3.2 Dimensão client machine

Nome	Tipo	Exemplo
Dispositivo	TEXTO	PC
Sistema Operativo	TEXTO	Windows 7
Browser	TEXTO	Chrome
Versão	TEXTO	46.0.2490.80 m
IP	TEXTO	111.111.111.111
Versão IP	TEXTO	Ipv4

Tabela 20 - Atributos da dimensão client machine

Hierarquias:

Nome	Hierarquias
BROWSER	**DISPOSITIVO => SISTEMA OPERATIVO => BROWSER => VERSÃO**
IP	**DISPOSITIVO => SISTEMA OPERATIVO => IP**
DISPOSITIVO	**DISPOSITIVO => SISTEMA OPERATIVO => IP**

Tabela 21 - Hierarquias da dimensão client machine

6.3.3 Factos

A tabela de factos tem como granularidade uma operação: O utilizador U com o dispositivo T realizou a operação O na data D, hora R e com um computador C.

A tabela de factos tem as seguintes medidas:

- **Valor:** Aplicável para operações que envolveram pagamentos. Nos restantes o valor é 0.
- **Repetições:** Conta o número de vezes em que a mesma operação ocorreu num curto espaço de tempo.

6.4 Data mart sistemas

Aplicando a modelação dimensional para o *data mart* dos pedidos, temos as dimensões data, relógio, utilizador, operação, canal, estado e a tabela de factos, conforme o diagrama abaixo:

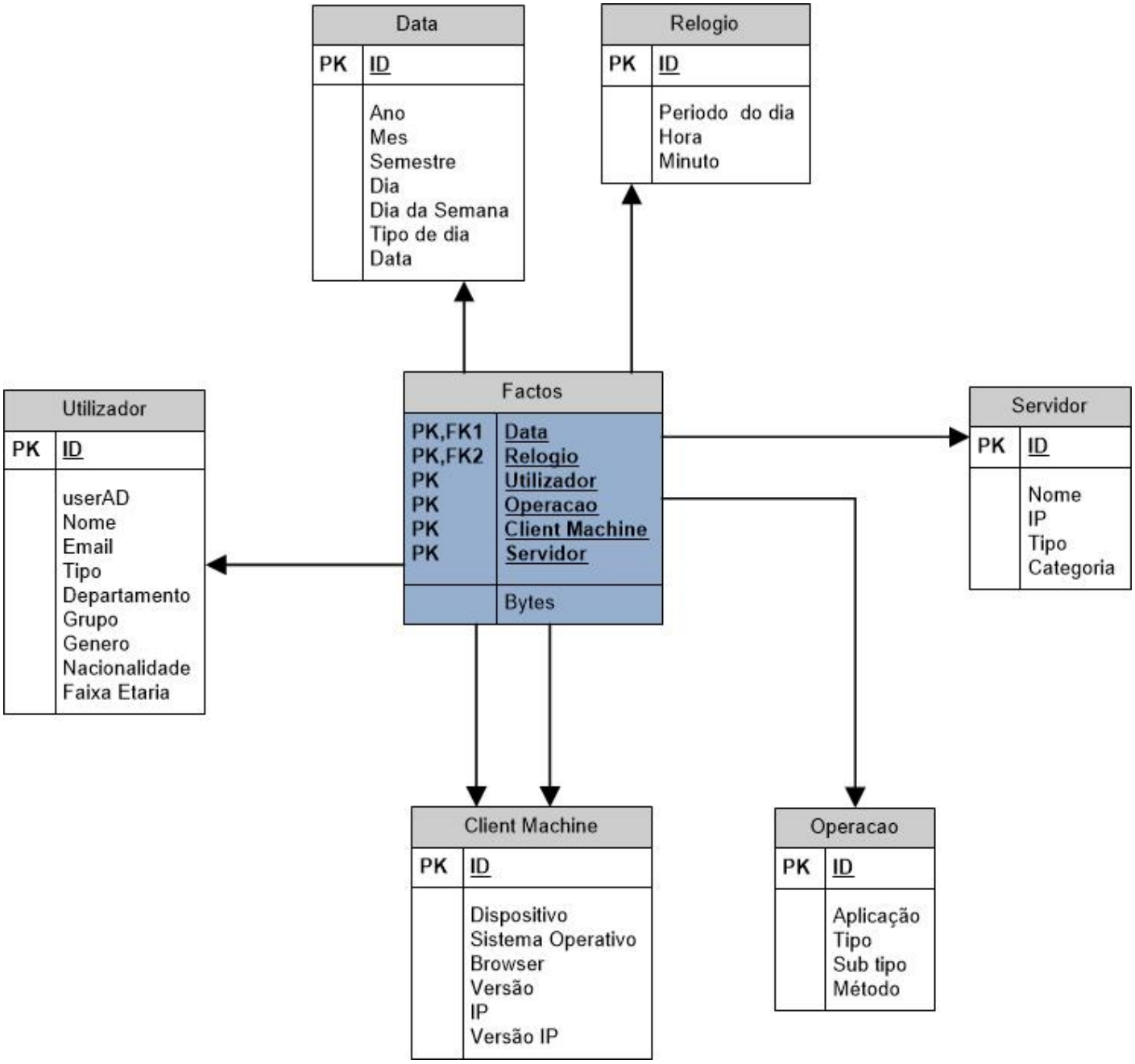


Figura 10 - Star Schema do data mart sistemas

6.4.1 Dimensões Data, Relógio, Utilizador, Operação, Client Machine

As dimensões Data, Relógio, Utilizador e Operação são conformadas com as equivalentes no *data mart* Pedidos. A dimensão Client Machine é conformada com a equivalente do *data mart* Aplicações.

6.4.2 Dimensão Servidor

Nome	Tipo	Exemplo
Nome	TEXTO	Web server 1
IP	TEXTO	123.123.123.123
Tipo	TEXTO	Apache
Categoria	TEXTO	Servidor web

Tabela 22 - Atributos da dimensão servidor

Hierarquias:

Nome	Hierarquias
	CATEGORIA => TIPO => NOME

Tabela 23 - Hierarquias da dimensão servidor

6.5 Data warehouse

6.5.1 Bus matrix

O *bus matrix* ajuda a entender quais as dimensões que são comuns entre os vários *data marts*. É especialmente importante para poder conformar as dimensões e planear a execução do ETL.

	Data	Tempo	Canal	Utilizador	Operação	Estado	Client Machine	Servidor
Pedidos	X	X	X	X	X	X		
Aplicações	X	X		X	X		X	
Sistemas	X	X		X	X		X	X

Tabela 24 - Bux matrix

6.5.2 *Fact Constellation*

Devido ao facto de as dimensões estarem conformadas, podemos representar todo o DW com os *data marts* Pedidos, Aplicações e Sistemas num único diagrama (ver próxima página):

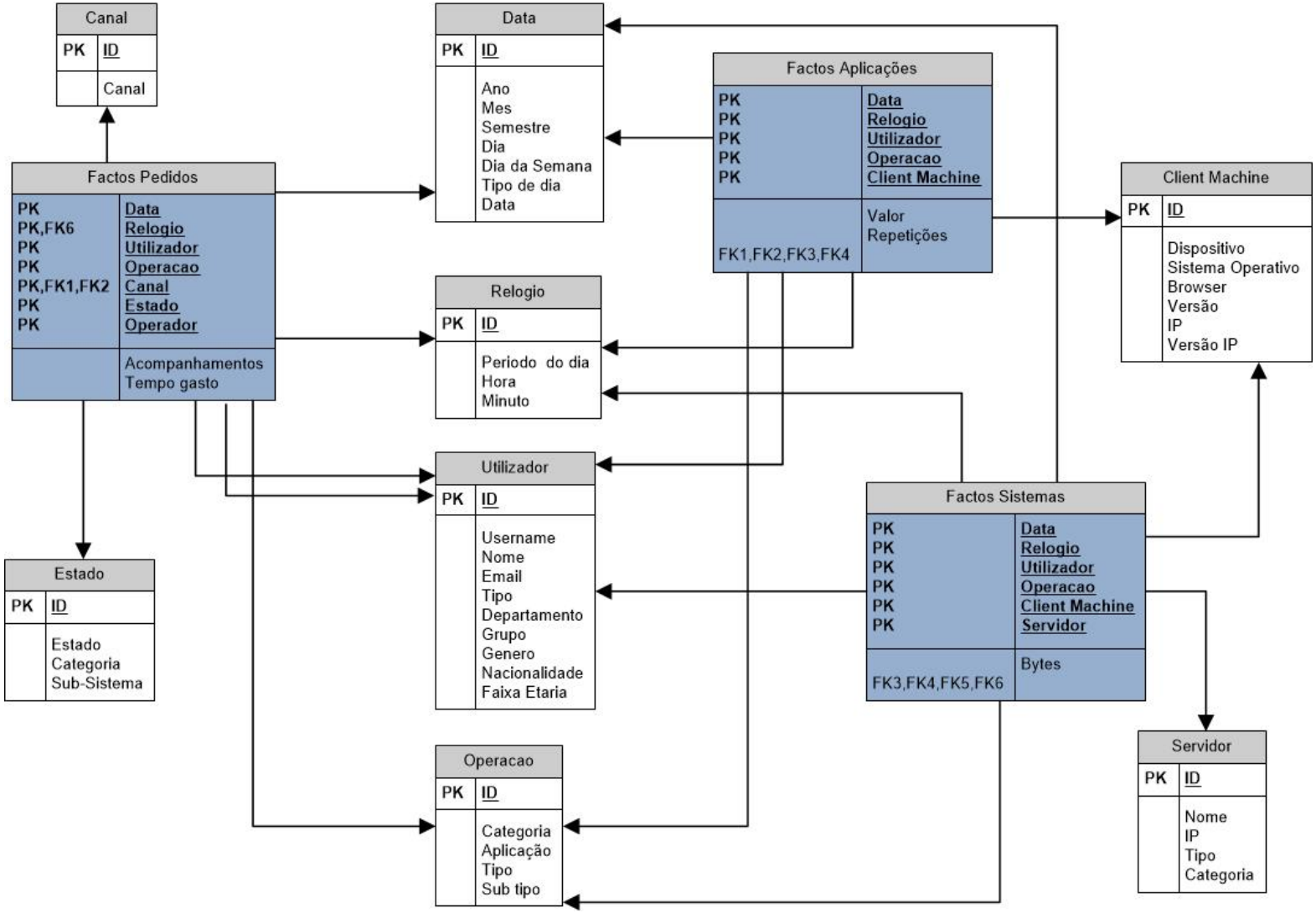


Figura 11- Fact Constellation

Capítulo 7 Desenho físico

O desenho físico [1,2,4,7] da DW tem um papel determinante na procura de um melhor aproveitamento de recursos e maior rapidez de acesso aos dados Nesta fase, a modelação dimensional é concretizada em tabelas de bases de dados, neste caso MySQL. Existem duas áreas distintas que vão ser desenhadas: a ***staging area*** e a ***presentation area***.

7.1.1 Presentation area

A *presentation area* será onde vão estar os dados da DW depois de extraídos, transformados e carregados no ETL. Esses dados são acedidos pelo cubo e por sua vez pelos utilizadores. As tabelas só devem ter os campos relevantes para o negócio e devem ter nomes percetíveis e amigáveis. É também aqui que se procede a otimizações para melhorar o desempenho no acesso aos dados recorrendo por exemplo a índices.

Existem dois tipos principais de índices os arvore B+ e os de função de dispersão (hash indexes). Os de árvore B+ são bons para pesquisas por igualdade ou por intervalo (*slice & dice*). Os com função de dispersão são superiores em pesquisas por igualdade, mas não servem para pesquisas por intervalo. São aplicados índices multi-atributo na tabela de factos ou nas tabelas de dimensões para campos que geralmente são apresentados em conjunto. Para a *presentation area* foram criados os seguintes índices:

Nome	**Campos**	**Tipo**	**Método**
CANAL			
canal	canal	Unique	BTREE
DATA			
Ano	Ano	Normal	BTREE
Ano mes	Ano,mes	Normal	BTREE
Tipo	Dia da semana, Tipo de dia	Normal	BTREE
ESTADO			

Categoria	Categoria	Normal	BTREE
Categoria estado	Categoria, Estado	Normal	BTREE
OPERACAO			
Tipos	Tipo de Operação, Sub-Tipo de Operação	Normal	BTREE
Categorias	Categoria	Normal	BTREE
Aplicações	Aplicação	Normal	BTREE
RELOGIO			
Período	Periodo do dia	Normal	BTREE
UTILIZADOR			
userAD	Username	Unique	BTREE
Tipo	Tipo de Utilizador	Normal	BTREE
Departamento	Departamento	Normal	BTREE
Nacionalidade	Nacionalidade	Normal	BTREE
Etaria	Faixa Etária	Normal	BTREE
Genero	Género	Normal	BTREE
FACTOS			
Tudo	dim_operacao_id, dim_data_id, dim_relogio_id, dim_utilizador_id, dim_operador_id, dim_estado_id, dim_canal_id	Normal	BTREE

Tabela 25 - Indicies aplicados às tabelas na presentation area

O desenho físico do modelo de dados será semelhante ao da modelação dimensional em que existe uma tabela de factos que liga às várias tabelas que representam as dimensões. Esta disposição chama-se modelo em estrela ou *star schema.*

Ao contrário dos sistemas operacionais que usam a 3NF com o intuído de diminuir redundâncias, numa base de dados de uma DW será precisamente o oposto. Existe redundância de dados, mas por outro lado o modelo é mais simples e fácil de interpretar. O desenho físico da *presentation area* terá a seguinte forma:

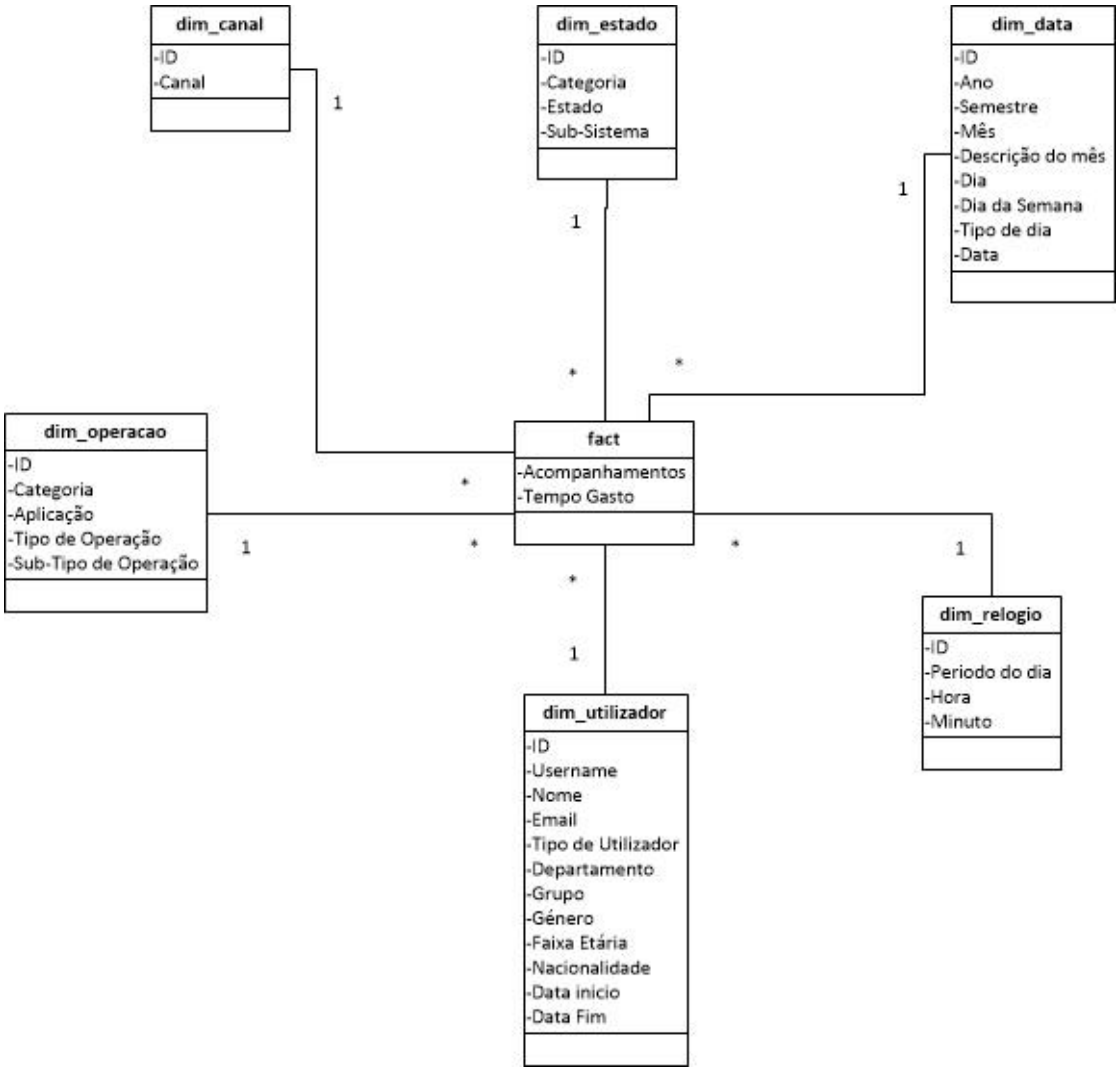


Figura 12 - Diagrama do modelo físico da presentation area

7.2 Staging area

A *staging area* é uma área intermédia entre os sistemas operacionais e a *presentation area*. Muitas vezes é questionada a necessidade da existência desta área intermédia, visto ser possível efetuar as transformações diretamente na *presentation area* e assim poupar uma instância de BD e espaço em disco. A não ser que a DW seja bastante rudimentar, o processo de ETL terá consolidação de dados de várias fontes, limpeza de campos, deteção de alterações. Efetuar diretamente estas operações na *presentation area*, pode tornar, nem que seja temporariamente, a base de dados incoerente e mostrar informação incorreta ou incompleta que possa induzir em erro os decisores. Esta é uma área de trabalho e nunca deverá ser acedida pelos utilizadores finais. As tabelas auxiliares que não fazendo parte da modelação ajudam no processo também estão situadas nesta área. Elas são:

Tabela	Descrição
dim_utilizador_TMP	Tabela que auxilia no ETL da dimensão Utilizador
dim_operacao_TMP	Tabela que auxilia no ETL da dimensão Operação
dim_estado_TMP	Tabela que auxilia no ETL da dimensão Estado
Processamentos_ETL	Regista as datas dos processamentos de ETL
tipo_report	Tipo de *report* a ser enviado
destinatarios	Endereço de email por onde vão ser enviados
envio_reports	Regista a data de envio de um report
Canal_template	Possui a lista dos canais da DW

Tabela 26 - Lista de tabelas auxiliares na staging area

As outras tabelas são similares às da *presentation area*, mas com a diferença de os nomes dos campos estarem mais orientados a facilitar *queries* de SQL evitando espaços e caracteres estranhos. Nestas tabelas, também vão existir mais atributos que ajudam no processo de ETL como por exemplo o campo “aux_estado” da tabela dim_estado que contém a chave primária no sistema operacional e ajudará no mapeamento entre a chave primaria no OLTP e a sua equivalente no DW.

O desenho físico do modelo de dados será da seguinte forma (ver próxima página):

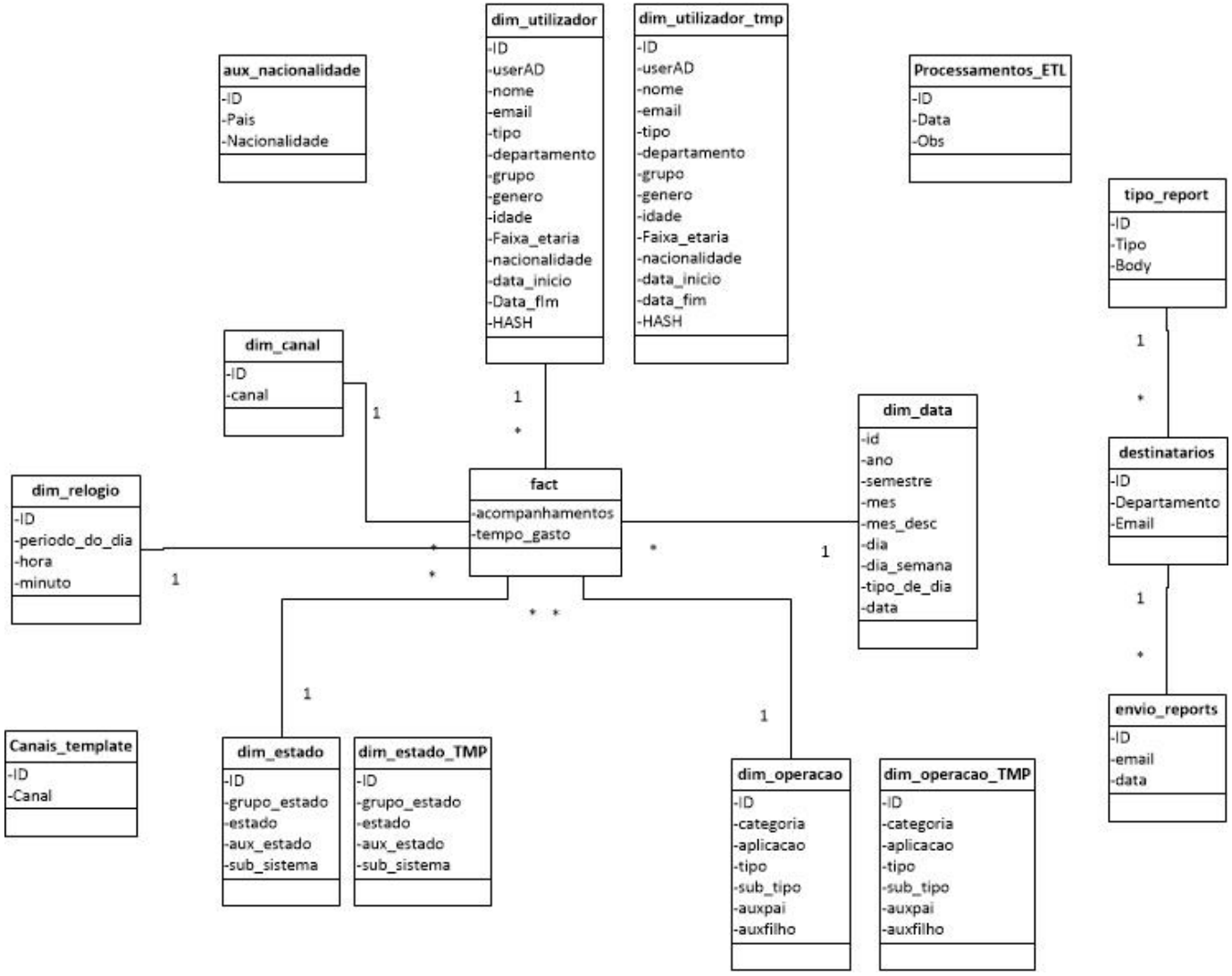


Figura 13 - Diagrama do modelo físico do staging area

Capítulo 8 - Processo de ETL

O objetivo do *Extract – Transform – Loading* (ETL) é extrair os dados das diversas fontes de dados dos sistemas operacionais, proceder a transformações de dados com o intuito de remover redundâncias, incoerências, normalização e carregar esses dados para um cubo de dados [1,2,3]. Este processo consome muito tempo e recursos, cerca de 70 % do esforço de construção da DW. Isto deve-se ao facto de haver muitas origens de dados distintas e ser necessário um tratamento *ad-hoc* para cada uma delas. Mesmo existindo ferramentas de *data integration* que ajudam no processo, cada fonte de dados tem os seus próprios erros e redundâncias. Também consome muito tempo a análise dessa fonte de dados em que torna difícil encontrar todos os erros. Em cada um dos passos será efetuado:

- **Extration:** Com o intuito de centralizar várias fontes de dados, analisar regras de integridade das colunas (ex. representação do mesmo utilizador em fontes diferentes) e outras regras de negócio. Nesta fase também se podem aplicar filtros de modo a descartar *a priori* dados que não têm utilidade. Isto será bastante importante no que toca ao tratamento dos *logs*. É também nesta fase que se deteta se ocorreu alterações dos dados e prepara-se o procedimento a tomar nesses casos.
- **Transformation:** Nesta fase, como o nome indica, os dados são transformados: fundir duplicados, correção de erros nos dados, mudar o valor de certos campos de forma a torna-los mais inteligíveis. Por exemplo, quanto o valor é vazio, substituir pela palavra “Desconhecido”. É nesta fase que também se trata exceções, como por exemplo, coisas que não existem na origem dos dados, mas que têm que ser acrescentadas ao sistema.
- **Load:** Depois de os dados estarem tratados são carregados no *data presentation* para serem utilizados. É nesta fase que se lida com a questão das dimensões de mudança lenta. É aqui que se calcula valores agregados e se preenche as hierarquias.

8.1 Arquitetura do ETL

No início do processo de desenvolvimento deste DW, foram criados pequenos programas em JAVA para ajudar a compreender o processo de ETL e proceder a alguns testes iniciais a ferramentas. Rapidamente compreendeu-se que existem muitos pequenos passos e que estes têm dependências entre si e que um processo tipo *batch* não serviria. Por exemplo, a tabela de factos só pode ser criada depois de todas as dimensões o terem sido. A ferramenta de *data intregation* da Pentaho chamada Kettle [11] permite criar esse fluxo de dependências de forma interativa. Existem dois tipos de processos no Kettle:

- ***Jobs*:** O *job* contém o fluxo do processo na sua globalidade, com várias ações e ligações entre si. A maior parte dessas ações vão ser *transformations*.
- ***Transformations*:** Cada transformação contém várias ações relacionadas com leitura, transformação e escrita de dados.

Devido ao facto de algumas transformações terem demasiada complexidade para poderem ser implementadas com os tipos de ações fornecidas pelo Kettle e acederem a fontes de dados externas como a base de dados dos alunos e funcionários, optou-se em retirar qualquer implementação do Kettle e centraliza-la num *bus* de serviços REST implementado em *cakePHP*. É utilizada a ação do Kettle "REST Client" que invocará um determinado *endpoint* efetuando uma determinada tarefa. CakePHP é uma *framework* PHP já usada nas aplicações *web* da UI. Para além de aproveitar as vantagens desta infraestrutura já estabelecida tais como, acesso a bases dados e *Active directory*, segurança e serviços já implementados, permite também que toda a equipa da área de desenvolvimento possam contribuir para a melhoria da implementação do ETL.

Deverão existir duas BD's distintas, a *data staging area* e a *data presentation area* conforme foi explicado no capítulo 7 – Desenho Físico.

8.2 Processos

Os processos utilizam as seguintes ações:

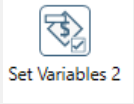
Ação	Descrição
	Início do processo no Job
	Ação do tipo “transformation” no job
	Execução de um script de SQL no job
	Envio de email no job
	Estabelece as variáveis globais na *transformation*
	Invoca um serviço REST na *transformation*
	Fim do processo no job

Tabela 27 - Tipos de ações do Kettle que foram utilizadas

O Job principal do Kettle terá a seguinte disposição (ver próxima página):

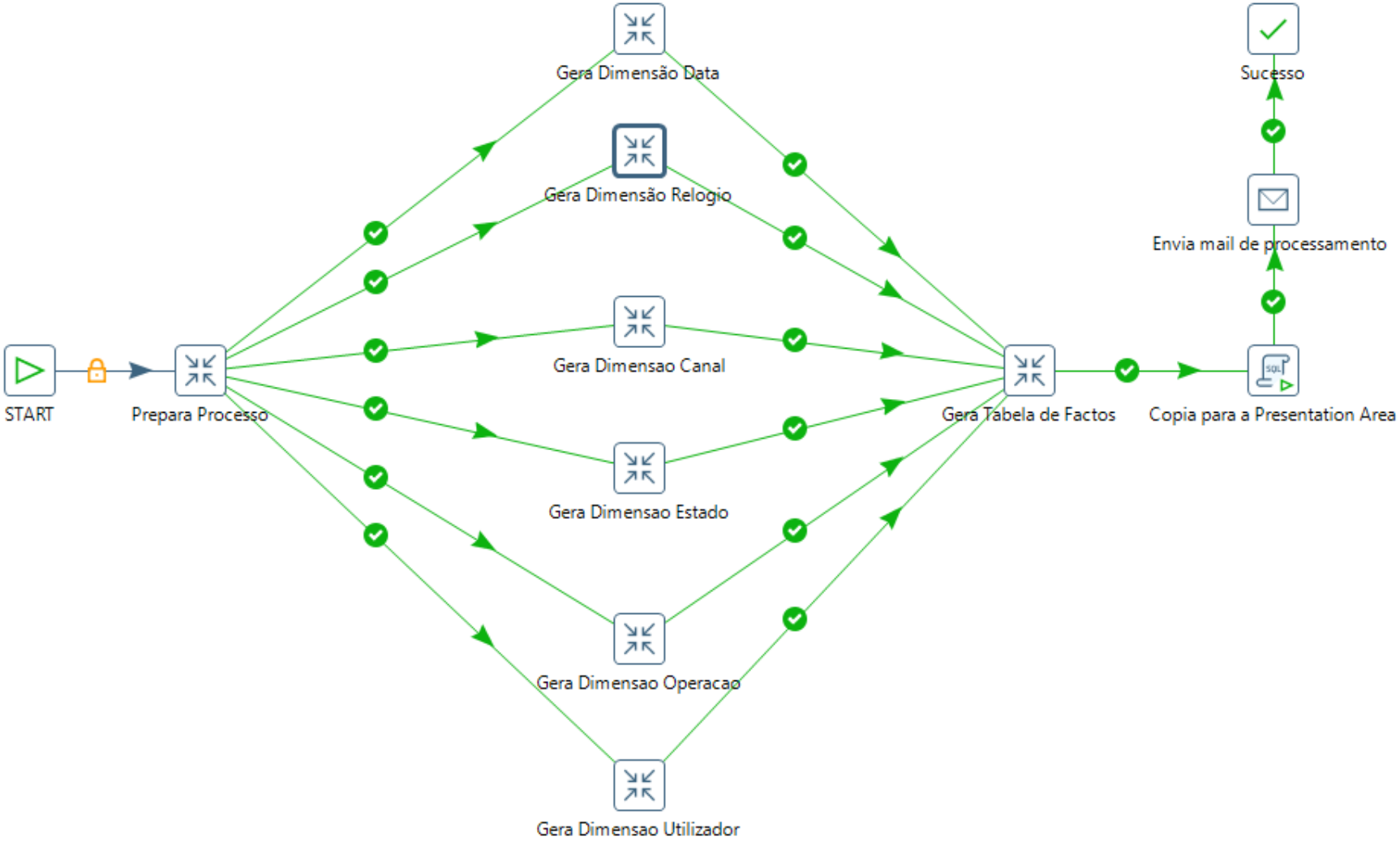


Figura 14 - Job do Kettle

8.2.1 Prepara processo

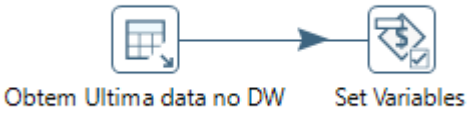


Figura 15 - Ações da transformação "Prepara Processo"

A tarefa “Obtem Ultima data da DW”, efectua uma *query* de SQL e obtém a data do último processamento. Caso as variáveis globais “dataInicio” e “dataFim” não estejam preenchidas, o valor “dataInicio” será a data do último processamento mais um dia e a data de fim como a data atual. Caso se pretenda correr o processo de uma forma *ad-hoc* pode-se especificar manualmente a data de início e de fim nas variáveis globais.

8.2.2 Dimensão data

Figura 16- Ações da transformação "Dimensão Data"

#	Acção	EndPoints
1	**Preparação**	\servicoDataWarehouse\DimDataController\Prepare
2	**Gera Dimensão Data**	\servicoDataWarehouse\DimDataController\geraDimData
3	**Preenche Semestre**	\servicoDataWarehouse\DimDataController\fillSemestre
4	**Preenche Feriados**	\servicoDataWarehouse\DimDataController\feriados

Tabela 28 - Endpoints para a dimensão data

Esta dimensão é gerada programaticamente não tendo nenhuma origem de dados. A primeira ação “preparação”, procede ao truncamento da tabela caso seja indicado. Neste momento esta ação está vazia, mas se por exemplo fosse acrescentado um novo atributo na dimensão, a tabela teria que ser truncada e gerada novamente.

No segundo passo “Gera Dimensão Data”, irá inserir na BD uma linha por data entre a data de início e a data de fim indicada nos passos anteriores. A chave primária é gerada a partir da data e tem o seguinte formato: <ano><mês><dia>. Os campos “ano” , “mês” e “dia” são preenchidos directamente a partir da data. O campo “mesDesc” é obtido através de uma função que mapeia o número do mês pela sua designação (exemplo: 01 => “Janeiro”). O campo “diaSemana” é preenchido com o auxílio das funções de manipulação de datas fornecidas pelo PHP. Neste caso devolve um inteiro de zero a seis sendo zero “Domingo” e seis “Sabado”. Tal como acontece com o campo “mesDesc” é necessária uma função que mapeie este número com a sua designação equivalente. O “tipo de dia” é preenchido a partir do dia de semana: Se for sábado ou domingo, preenche como “Fim de semana”, caso contrário como dia de Semana.

No terceiro passo, o campo “semestre” será atualizado, acedendo a um serviço do SIGES em que obtém o intervalo de datas dos semestres e preenche respectivamente “1º Semestre” ou “2º Semestre”

Finalmente o quarto passo, atualiza o campo “tipo de dia” para “Feriado” para os dias que estão assinalados como tal na aplicação de marcação de ferias dos funcionarios da FCUL.

8.2.3 Dimensão relógio

Figura 17 - Ações da transformação "Dimensão Relógio"

#	Acção	EndPoints
1	**Preparação**	\servicoDataWarehouse\DimTimeController\Prepare
2	**Gera Dimensão Relogio**	\servicoDataWarehouse\DimTimeController\geraDimTime

Tabela 29 - Endpoints da dimensão relógio

Esta dimensão é gerada programaticamente. A primeira acção “preparação”, procede ao truncate da tabela tal como foi referido na dimensão Data.

No segundo passo “gera Dimensão Relogio”. Irá gerar uma linha para todos os minutos de um dia , ou seja 1440. A chave primária tem o seguinte formato <hora><minuto>. O campo “periodo do dia” é obtido a partir da hora e segue a seguinte regra:

Regra	Valor
Hora]6-13]	Mannhã
Hora]13-20[	Tarde
Hora [20-6]	Noite

Tabela 30 - Regras para a atribuição do periodo do dia

8.2.4 Dimensão canal

Figura 18 - Ações da transformação "Dimensão Canal"

#	Acção	EndPoints
1	**Preparação**	\servicoDataWarehouse\DimCanalController\Prepare
2	**Gera Dimensão Canal**	\servicoDataWarehouse\DimCanalController\geraDimCanal

Tabela 31 - Endpoints da Dimensão Canal

Esta dimensão tem poucas opções possiveis e vão mudar pouco ao longo do tempo e por isso decidiu-se criar uma tabela auxiliar chama “canal_template” na *staging area* que será a fonte de dados.

A primeira acção “preparação”, procede ao truncate da tabela (caso seja indicado).

O segundo passo “Gera Dimensão Canal” copia os dados da tabela “canal_ template” para a tabela “dim_canal”.

8.2.5 Dimensão estado

Figura 19 - Ações da transformação "Dimensão Estado"

#	Acção	EndPoints
1	**Preparação**	\servicoDataWarehouse\DimEstadoController\Prepare
2	**Insere auxiliar**	\servicoDataWarehouse\DimEstadoController\insertAux
3	**Gera Dimensão Estado**	\servicoDataWarehouse\DimEstadoController\geraDimEstado

Tabela 32 - Endpoints da dimensão estado

Na primeria fase “preparação” a tabela dim_estado_aux é truncada.

No passo dois “insere auxiliar”, os estados provenientes das aplicações das várias origens de dados são inseridos em”Dim_estado_TMP”. O campo “estado_id” tem a chave primária proveniente da tabela de origem dos dados original, ajudando a distinguir se o estado foi modificado ou se é novo.

No passo três “Gera Dimensão Estado”, vai percorrer todas as linhas da tabela “dim_estado_TMP”. Vai comparar os campos da tabela dim_estado_aux com a tabela dim_estado através do cruzamento da origem de dados e do campo “estado_id”. Existem três possibilidades:

1. O estado não sofreu alterações
2. O estado foi alterado e procede-se a uma alteração do tipo 1 sobrepondo o valor dos campos em “dim_estado_TMP” para dim_estado.
3. O estado é novo, ou seja o estado_id existe em “dim_estado_TMP” , mas ainda não existe em dim_estado. É inserido o novo estado em dim_estado

Era suposto criar um automatismo para preencher o campo “grupo estado”, mas provou-se ser complexo senão impossivel pegar nos campos que existem à disposição e adivinhar qual o valor do campo a tomar. O estados provenientes dos sistemas operacionais são muito variados, estão em portugues e em inglês e existem estados de dificil catalogação. Por essa razão, quando um novo estado é criado, este campo é preenchido como “a definir” e terá que ser modificado manualmente.

8.2.6 Dimensão operação

Figura 20 - Ações da transformação da "Dimensão Operação"

#	Acção	EndPoints
1	**Preparação**	\servicoDataWarehouse\DimOperacaoController\Prepare
2	**Insere auxiliar**	\servicoDataWarehouse\DimOperacaoController\insertAux
3	**Gera Dimensão Operacao**	\servicoDataWarehouse\DimOperacaoController\geraDimOperacao

Tabela 33 - Endpoints da Dimensão Operação

Na primeira fase “preparação” a tabela “dim_operacao_TMP” é truncada.

No passo dois “insere auxiliar”, as operações provenientes das aplicações das várias origens de dados são inseridos em “Dim_operacao_TMP”. Os campos “aux_pai” e “aux_filho” têm a chave primária proveniente da tabela de origem dos dados original e de quem descendia (se aplicável), ajudando a distinguir se a operação foi modificada ou se é nova. Nos caso em que a operação só tem um nível (não depende de outra) só é preenchido o campo “aux_pai”e “aux_filho” fica preenchido com “0” (zero).

No passo três “Gera Dimensão Operacao” vai percorrer todas as linhas da tabela dim_ operacao _aux. Vai comparar os campos da tabela “dim_operacao_TMP” com a tabela “dim_operacao” através do cruzamento da origem de dados e dos campos “aux_pai” e “aux_filho”. Existem três possibilidades:

1. A operação não sofreu alterações
2. A operação foi alterada e procede-se a uma alteração do tipo 1 sobrepondo o valor dos campos em dim_ operacao _TMP para dim_ operacao.
3. A operação é nova, e é inserida em dim_ operacao

8.2.7 Dimensão utilizador

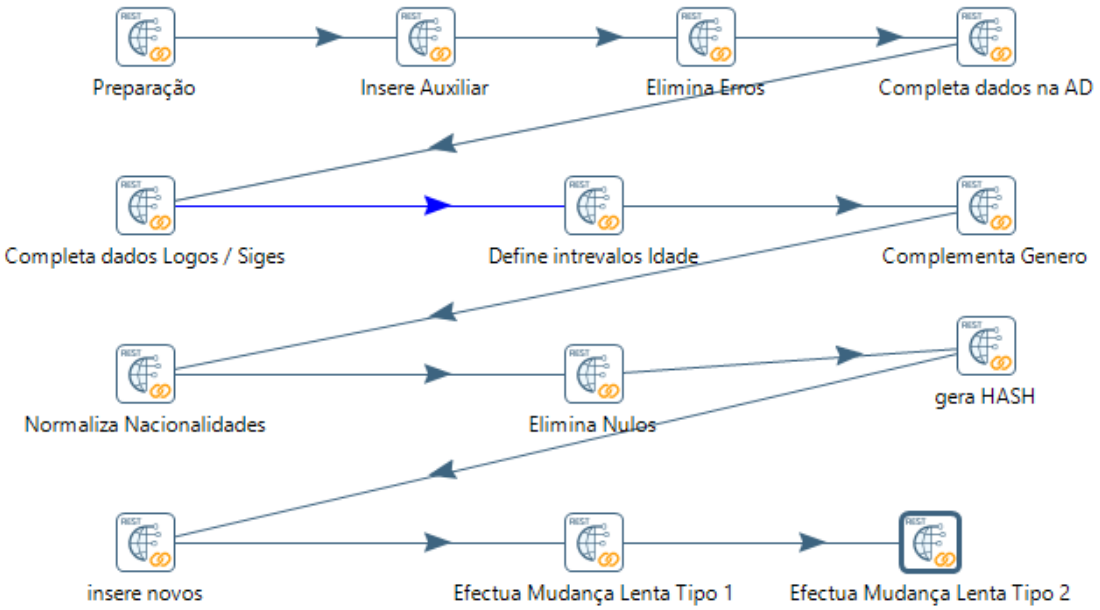


Figura 21 - Ações da transformação "Dimensão Utilizador"

#	Acção	EndPoints
1	**Preparação**	\servicoDataWarehouse\DimUtilizadorController\Prepare
2	**Insere auxiliar**	\servicoDataWarehouse\DimUtilizadorController\insertAux
3	**Elimina Erros**	\servicoDataWarehouse\DimUtilizadorController\limpa
4	**Completa Dados AD**	\servicoDataWarehouse\DimUtilizadorController\fillAD
5	**Completa Dados LOGOS / SIGES**	\servicoDataWarehouse\DimUtilizadorController\fillLogosSiges
6	**Define Intrevalos de**	\servicoDataWarehouse\DimUtilizadorController\setIdade

	Idade	
7	**Complementa Genero**	\servicoDataWarehouse\DimUtilizadorController\GeneroByOthers
8	**Normaliza Nacionalidades**	\servicoDataWarehouse\DimUtilizadorController\normNacionalidade
9	**Elimina Nulos**	\servicoDataWarehouse\DimUtilizadorController\noNull
10	**Gera HASH**	\servicoDataWarehouse\DimUtilizadorController\hash
11	**InsereNovos**	\servicoDataWarehouse\DimUtilizadorController\insertNew
12	**Efectua Mudança Lenta Tipo 1**	\servicoDataWarehouse\DimUtilizadorController\updateT1
13	**Efectua Mudança Lenta Tipo 2**	\servicoDataWarehouse\DimUtilizadorController\updateT2

Tabela 34 - Endpoints da "Dimensão Utilizador"

Na primeira fase “preparação” a tabela “dim_utilizador_TMP” é truncada.

Na fase dois “insere auxiliar”, todos os utilizadores que constam dos sistemas operacionais são inseridos na tabela dim_utilizador_TMP.

Na fase três “Elimina Erros”, são solucionados os problemas com o campo “userAd”. Sendo este uma chave susbstituta que representa o utilizador e o ponto de partida para os passos seguintes em que os seus dados vão ser complementados, é importante corrigir incorreções e descartar os inúteis. As verificações são as seguintes:

- Transformar os casos em que o campo “userAD” é um endereço de *email* da FCUL (“@fc.ul.pt”, “@alunos.fc.ul.pt” e “@ciencias.ulisboa.pt”) retirando o seu sufixo. Exemplo: rjsimoes@fc.ul.pt => rjsimoes.
- Descartar os *usernames* que não são *emails* válidos ou que são nulos ou que têm menos de três caracteres.
- Descartar utilizadores cujo o email não pertence à FCUL (“@gmail”,”@hotmail”, etc). Estes utilizadores vão ser posteriormente associados a um utilizador especial chamado “EXTERNO”.

Na fase quatro “Completa dados na AD”, para cada “username” irá consultar os dados no *active directory* de onde irá preencher os campos “nome” e “email” e ainda determinar se o utilizador é docente, funcionário ou aluno preenchendo essa informação no campo “tipo”. O campo “Grupo” é preenchido se o utilizador pertencer aos grupos “Operador” ou “Técnico” no *active directory*. Caso não pertença nem a um nem a outro será preenchido “Nenhum”.

Na fase cinco “Complementa dados Logos / SIGES”, vai complementar os dados “departamento”, “genero” , “idade” e “nacionalidade”. Se o utilizador for um aluno consulta a BD SIGES, caso contrário consulta o sistema LOGOS. A idade é calculada a partir da data de nascimento do utilizador. No LOGOS existem muitos utilizadores em que a data de nascimento é “01-01-1970”. Sendo uma data por omissão e sendo dificil de distinguir os verdadeiros utilizadores que nasceram a “01-01-1970”, optou-se para estes casos colocar “Desconhecido”.

Na fase seis “Define intervalos de idade” vai preencher o campo “Faixa Etária” determinando em que intervalo da idade calculada anteriormente pertence.

Na fase sete “Complementa género”, como existem bastantes casos em que a informação do género estava omissa no SIGES / LOGOS, irá tentar preencher este campo comparando o primeiro nome do utilizador com outros primeiros nomes de utilizadores que tenham o campo género preenchido.

Na fase oito “Normaliza Nacionalidades”, o campo “nacionalidade” é uniformizado devido ao facto de no sistema SIGES ser devolvido o país em maiúsculas (exemplo: “PORTUGAL”) e no LOGOS a nacionalidade em minusculas (“Portuguesa”). Para tal existe uma tabela auxiliar aux_nacionalidades na *staging area* que permirá fazer o mapeamento correcto e proceder à alteração.

Na fase nove “Elimina Nulos” todos os campos da tabela “dim_utilizador_TMP” que estejam vazios ou nulos são preenchidos com a palavra “Desconhecido”

Na fase dez “gera Hash” é preenchido o campo “HASH” fazendo uma síntese de uma *string* concatenada com todos os campos do utilizador. Esta sintese é feita através da função “hash” do PHP. Este campo será utilizado posteriormente para verificar se existem alterações nos dados dos utilizadores.

Na fase onze “Insere novos” os utilizadores que existem em dim_utilizador_TMP e que ainda não em dim_utilizador são inseridos. Esta verificação é feita através do *username* já que é uma chave substituta.

Na fase doze “Efectua mudança lenta Tipo 1” , comparando o HASH da tabela dim_utilizador_TMP e dim_utilizador vai verificar os utilizadores que sofreram alterações. Os campos “nome”, “email”,“género”, “Faixa Etária” e “Nacionalidade” são

alterados segundo os trâmites do tipo 1, esmagando a informação que existia anteriormente.

Na fase treze “Efectua mudança lenta Tipo 2” , comparando o HASH da tabela dim_utilizador_TMP e dim_utilizador, vai verificar os utilizadores que sofreram alterações. Os campos “Departamento”, “Tipo de utilizador” e “grupo” vão ser modificados segundo os trâmites de uma alteração do tipo 2. É criada uma nova linha com as alterações. A data atual é colocada no campo “dataFim” da linha antiga e no campo “dataInicio” da nova linha. O campo “dataFim” da nova linha estará vazio, indicando que é esta a ativa.

8.2.8 Factos

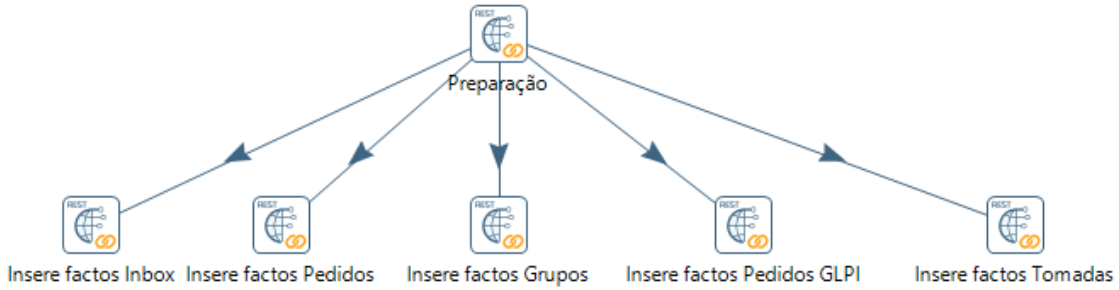


Figura 22 - Ações da transformação "Factos"

#	Acção	EndPoints
1	**Preparação**	\servicoDataWarehouse\FactosController \Prepare
2	**Insere Factos Inbox**	\servicoDataWarehouse\FactosController\insertInbox
3	**Insere Factos Pedidos**	\servicoDataWarehouse\FactosController\insertPedidos
4	**Insere Factos Grupos**	\servicoDataWarehouse\FactosController\insertGrupos
5	**Insere Factos Pedidos GLPI**	\servicoDataWarehouse\FactosController\insertPedidosGLPI

6	**Insere Factos Tomadas**	\servicoDataWarehouse\FactosController\insertTomadas

Tabela 35 - Endpoints das factos

Na primeira operação “Prepare” são executadas ações prévias. Neste momento não existe nenhuma, mas está prevista a transição para o futuro.

As operações “Insere Factos Inbox”, “Insere Factos Pedidos”, “Insere Factos Grupos”, “Insere Factos Pedidos GLPI” e “Insere Factos Tomadas” vão ser executadas em paralelo. Cada uma irá à sua respectiva fonte de dados obter os factos a ser inseridos através de uma *query* de SQL. Haverá métodos auxiliares que ajudaram a converter as chaves primárias das entidades nos sistemas operacionais para as chaves substitutas nas respectivas dimensões:

- **Utilizador:** pelo atributo “username” e caso já tenha ocorrido uma mudança de tipo 2, pelo que estiver ativo (com dataFim a null).
- **Data e Relogio:** pela alteração da formatação (ver capitulo 8.2.2)
- **Canal:** Existe uma função que faz um mapeamento directo.
- **Operação:** pelos atributos “Aplicação”, “auxpai” e “auxfilho”
- **Estado:** pelos atributos “sub_sistema” e “aux_estado”

Relativamente às medidas:

Operação	Tempo Gasto	Acompanhamentos
Pedidos	O que vem no OLTP	O que vem no OLTP
Pedidos GLPI	O que vem no OLTP	O que vem no OLTP
Inbox	0	0
Gestão Grupos	10	1
Tomadas:		
Activação	30	1
Desactivação	30	1
Alteração de Equipamento	15	0
Mover equipamento	30	1

Tabela 36 - Atribuição do tempo gasto e Acompanhamentos dependendo do tipo de operação

No futuro as ligações para “Insere Factos Inbox” e “Insere Factos Pedidos” vão ser desativadas pelo facto destas aplicações terem sido descontinuadas e os sistemas operacionais não irem produzir novos dados.

8.2.9 Copia para a *presentation area*

Existe um *script* de base de dados que copia as alterações efetuadas na *staging area* para a *presentation area*. Neste momento as tabelas da *presentation area* são truncadas e populadas na íntegra a partir das equivalentes na *staging area*, porque dado o volume de dados existentes por enquanto não justifica efetuar um *script* mais complexo que só copie as diferenças.

8.2.10 Envia e-mail de Processamento

Envia um *email* a indicar que o processo decorreu.

8.3 Operacionalizar o carregamento de dados do sistema ETL

Neste momento, com o volume de dados existente, o processo de ETL demora entre 20 e 25 minutos. Por essa razão optou-se por executá-lo diariamente de madrugada, permitindo aos utilizadores terem dados frescos até o dia anterior.

O processo de ETL poderá ser executado de duas formas:

- **Automaticamente**: É definido um período de tempo em que o processo é desencadeado e este ocorre sem intervenção humana.
- ***Ad-hoc***: Caso seja preciso forçar uma execução, quer para obter dados atualizados no momento, quer porque tenha ocorrido um problema ou seja preciso correr o processo novamente

8.3.1 Automaticamente:

Para que o ETL corra periodicamente e de forma automática deverá ser executada a seguinte linha de comandos:

```
kitchen.sh -file=/<raiz_do_kettle>/DM Pedidos.kjb --level=Minimal >> /<raiz_do_kettle>/LOG/trans.log
```

“DM Pedidos.kjb” é o ficheiro principal com o Job e todo o output é direcionado para o ficheiro “/LOG/trans.log”. A execução periódica será feita em modo *cron* e invocará esta linha de comando.

8.3.2 Ad-hoc

Para fazer uma execução *ad-hoc*, o *Job* "DM Pedidos" deverá ser aberto no Kettle, ir ao menu "Edit" > "Settings", ao tabulador "Parameters" e preencher as variáveis "DataInicio" e "DataFim". Depois é correr o Job no ícone *play*.

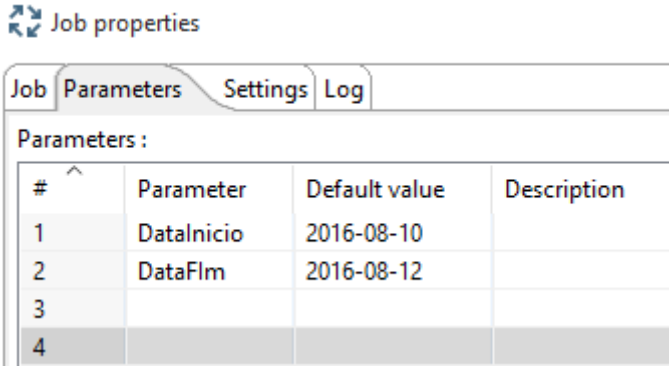


Figura 23 - Parâmetros "DataInicio" e "DataFIm" no Kettle

Efetuar o *job* no Kettle também tem a vantagem de ativar / desativar certas ligações, caso o operador só queira executar uma determinada parte do *Job*.

Capítulo 9 Definição do cubo de dados

O Mondrian [6,10] é um projeto *open source* inspirado num produto não comercial: o Microsoft Analysis Services. Este produto contém a linguagem MDX [8,13] como extensão do SQL para *business intelligence* e XMLA baseado em XML como interface de comunicação. O Mondrian é um *engine* OLAP implementado em JAVA e que corre num *web server* como o Tomcat ou o Jboss.

O Mondrian permite definir *schemas* dos cubos de dados que no fundo é um ficheiro de XML que determina a estrutura lógica de como os dados das tabelas de base de dados relacional aparecem no cubo. Esta camada permite a criação de campos virtuais que não existem no modelo físico, como por exemplo medidas calculadas. Permite também a possibilidade de alterar nomes dos campos e definir hierarquias. Pode ser vantajoso numa situação em que com a mesma DW como ponto de partida, pode-se criar vários cubos orientados às necessidades de diferentes utilizadores, aproximando-os do vocabulário e estrutura de dados que estão mais habituados.

O Mondrian contém muitos projetos associados que cresceram em torno dele como por exemplo o Jpivot, Olap4J, Ctools, etc.

9.1 Instalação

Sendo um projeto *open source* existem várias *releases* e *branches* (ver no gitHub https://github.com/pentaho/mondrian). As versões que contêm o Jpivot já integrado e o ficheiro "mondian.war" são as melhores como será explicado adiante.

O Mondrian pode ser *deployed* num servidor aplicacional ou ser utilizado através de outras aplicações *web*. A forma mais rápida e comoda é colocar o ficheiro "mondrian.war" existente na *release* na pasta "webapps" do *Tomcat* e fazer *restart* ao serviço. A integração com o Jpivot permite verificar se a instalação foi bem sucedida ou acedendo ao endreço (http://<servidor>:8080/mondrian/). Deverá apresentar uma lista com várias opções de teste baseadas no DW de demostração chamado "foodMart".

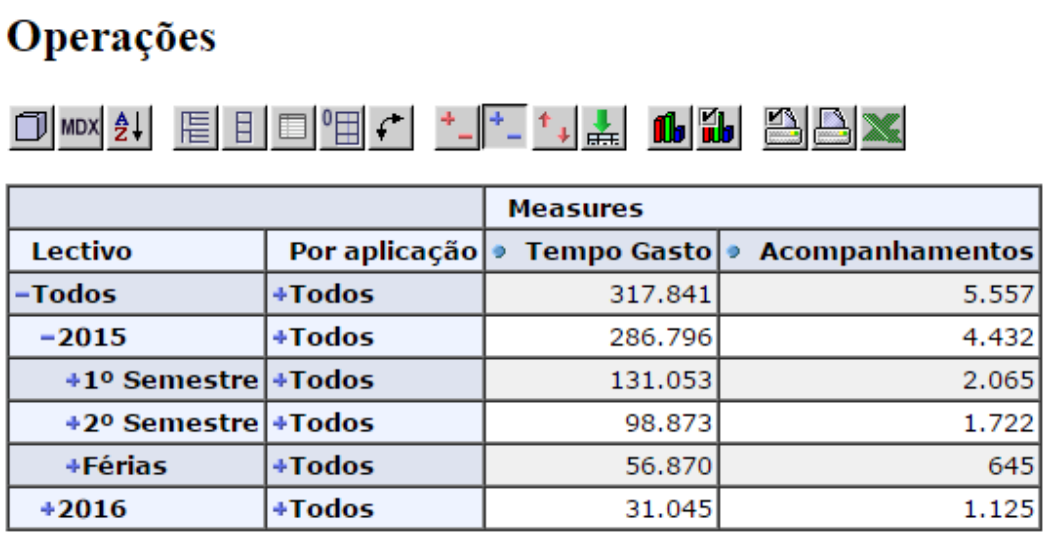


Figura 24 - JPivot

Como o Mondrian irá aceder a uma base de dados em MySql, é preciso fazer o *download* do *driver* "com.mysql.jdbc.driver" e colocar o ficheiro "Mysql.jar" na diretoria "lib" do projecto.

O próximo passo será criar o schema em XML.

9.2 Schemas do cubo de dados

A definição do *schema* segue uma notação própria especificada na documentação. São definidas as dimensões com os respetivos atributos e suas hierarquias. Estes atributos mapeiam com os campos de BD. É definida a tabela de factos e medidas. Pode-se criar mais medidas com tipos de agregação diferente

Estes s*chemas* em XML têm uma sintaxe muito precisa e as mensagens de erro são muito opacas. Felizmente a Pentaho desenvolveu uma ferramenta para ajudar a construir estes ficheiros: o *schema workbench*. Para tirar total proveito desta ferramenta, pode-se preencher os dados de conexão à base de dados acedendo ao item de menu "Tools". Esta opção permite mapear corretamente o *schema* com os campos da BD no DW. De referir que é comum ocorrer um problema quando se executa este programa (mondrian.sh para Linux e mondrian.bat para Windows) relacionado com o driver JDBC do Mysql. Para o resolver ou se cria uma diretoria "drivers" e se coloca o ficheiro do driver "Mysql.jar" ou corrige-se o script.

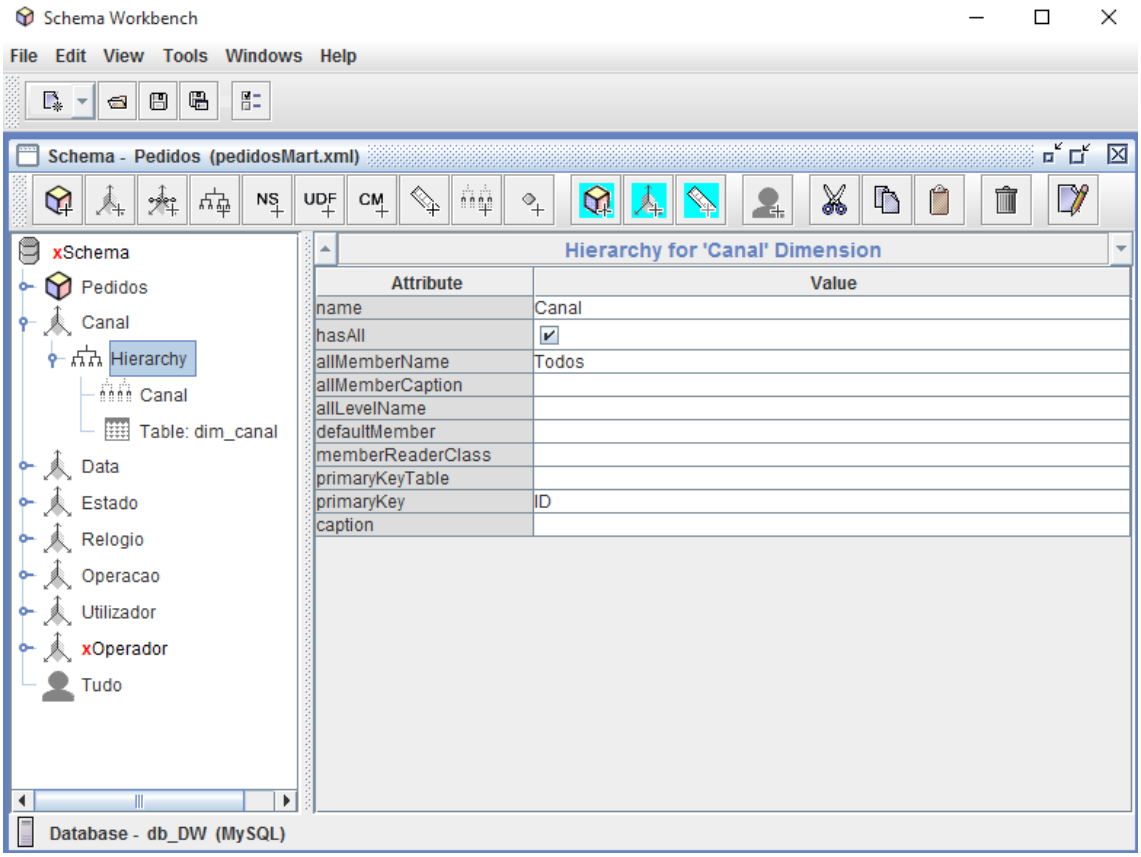


Figura 25 - Schema Workbench

Apesar de esta ferramenta ajudar bastante na construção do *schema*, não efetua qualquer tipo de validação de estrutura e sintaxe do documento, sendo bastante fácil de cometer erros que depois são difíceis de depurar. Para mitigar estes problemas, é aconselhável inserir uma dimensão de cada vez e começar com as mais simples. A construção é explicada com algum detalhe na documentação da ferramenta e também existem alguns vídeos elucidativos no *youtube* que auxiliam nesta tarefa.

Ao fazer "new schema" irá criar um ficheiro só com o elemento "schema" em que se pode dar um nome mais inteligível. O primeiro passo será criar as dimensões. Em cada uma existem várias opções, mas as mais importantes consistem em associar a tabela de BD, os campos e definir as hierarquias. Para cada hierarquia define-se os níveis "Levels" em que se mapeia com o respetivo campo na BD. A ordem com que esses níveis são criados vai definir a sequência da hierarquia. A um nível se pode adicionar "properties" que seriam campos informativos. Por exemplo, numa hierarquia da dimensão utilizador Departamento => Grupo => Utilizador, faz sentido acrescentar os campos "Nacionalidade","Género", etc como *properties* no nível "utilizador". O problema é que na maioria das ferramentas analíticas que foram usadas estas *properties* são ignoradas. A resolução deste problema passou por criar mais hierarquia com esses campos.

O próximo passo é criar um cubo dentro do *schema* associando a tabela de factos e criar dimensões associando-as às dimensões criadas anteriormente. Pode parecer uma redundância, mas para um universo em que existam vários cubos que usam as mesmas dimensões conformadas (e tabelas de factos diferentes) só é preciso defini-las uma única vez. Finalmente, define-se no cubo as medidas, associando o campo da tabela de factos e o tipo de agregação (*sum, avg, count*). Foi criado uma medida chamada "contagem" que não mapeia diretamente a nenhum campo da tabela de factos na BD fazendo um *count* das ocorrências. Também é possível criar dimensões calculadas e ainda formatar valores. Para a medida "tempo gasto" que é minutos, tentou-se formatar da forma "x h y m" no *schema*, mas sem sucesso. Esta formatação é possível de fazer diretamente na *query* de MDX (ver Anexo F– Formatar tempo gasto no MDX)

9.3 Configuração

Depois de criar os *schemas* é necessário definir os *data sources* que ligam os *schemas* ao DW físico com uma ligação de BD. É útil colocar a funcionar o Jpivot para fazer testes e verificar se o *schema* em XML está correto. Deve-se copiar o ficheiro do schema para /WEB-INF/queries/ e alterar os ficheiros /WEB-INF/queries/Mondrian.jsp e /WEB-INF/queries/mondrianXMLA.jsp:

Atributos de jp:mondrianQuery	**Valor**
jdbcDriver	Com.mysql.jdbc.Driver
jdbcUrl	Jdbc:mysql://<servidorBD>/<database>?user=<username>&password=<pass>
catalogUri	/WEB-INF/queries/<ficheiro_schema>.xml

Tabela 37- Atributos a preencher em jp:mondrianQuery

Dentro do elemento <jp:mondrianQuery> deverá estar uma *query* MDX válida. No contexto deste projeto poderia ser:

```
<jp:mondrianQuery id="query01" jdbcDriver="com.mysql.jdbc.Driver"
jdbcUrl="jdbc:mysql://localhost/db_DW?user=xxxxx&password=yyyyy"
catalogUri="/WEB-INF/queries/pedidosMart.xml"    connectionPooling="false">
select {[Measures].[Tempo Gasto], [Measures].[Acompanhamentos]} ON COLUMNS,
                                Hierarchize({([Data].[Todos], [Operacao].[Todos])}) ON
ROWS
from [Pedidos]
</jp:mondrianQuery>
```

Acedendo ao endereço:

http://<servidor>:8080/mondrian/testpage.jsp?query=mondrian

Deverá aparecer uma página da ferramenta Jpivot e a *query* definida anteriormente como a de omissão. Este é o ponto de partida para proceder a eventuais afinações do ficheiro do *schema* até estar tudo em condições de passar a usar outras ferramentas.

Capítulo 10 Ferramentas analíticas

Foram usadas as seguintes ferramentas analíticas: Jpivot, Saiku, Excel, relatórios em cakePHP / DOMPDF e o WEKA.

10.1 JPivot

Tendo sido bem configurado conforme o expresso no capítulo 9 - Definição do Cubo de dados, o Jpivot [16] poderá ser acedido através de seguinte endereço:

http://<servidor>:8080/mondrian/testpage.jsp?query=mondrian

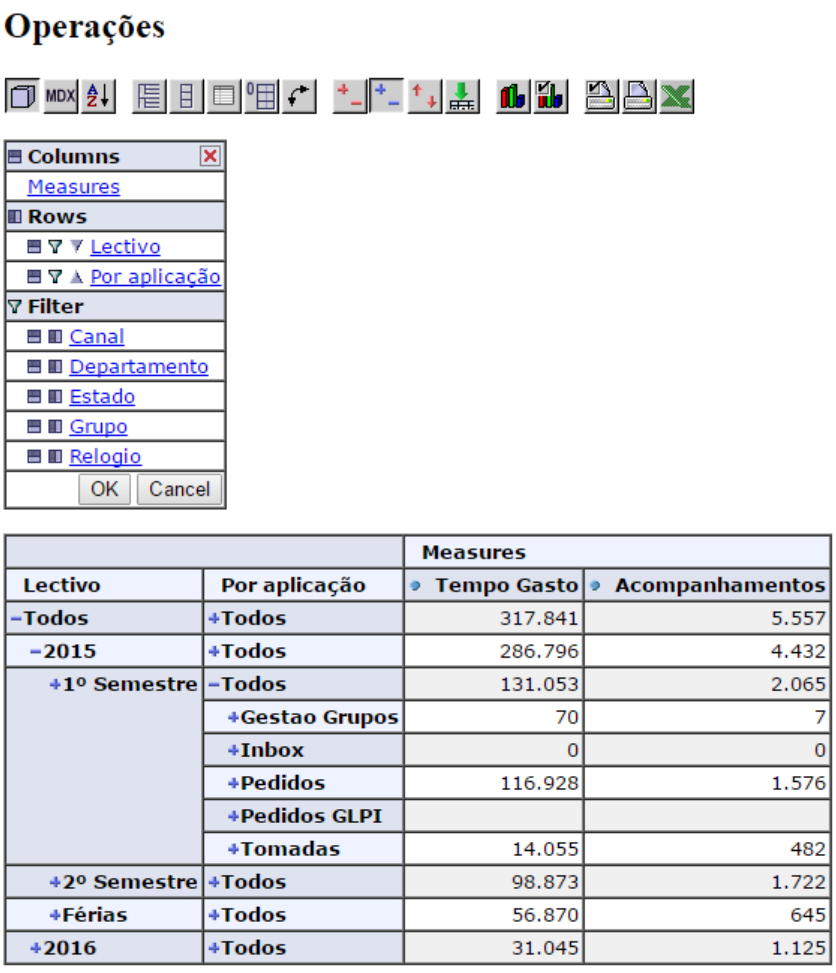


Figura 26 - Jpivot com as opções do cubo abertas

Como foi referido anteriormente, esta ferramenta é muito útil no processo de construção do cubo OLAP porque permite uma visualização rápida das alterações efetuadas.

A tabela é resultado da *query* de MDX colocada em "mondiran.jsp" (ver capítulo 9 - Definição do Cubo de dados). Os sinais de "+" e de "-" efetuam respetivamente *drill down* e *roll-up*. Para alterar os dados visualizados existem duas opções:

- Acedendo ao ícone do cubo, em que é mostrado uma tabela com "Columns" onde estão as medidas, "Rows" onde estão hierarquias e "Filter" onde se pode filtrar os resultados colocando condições nos campos das várias dimensões.
- Alterando a *query* de MDX no ícone "MDX".

O JPivot possui outras funcionalidades tais como gerar gráficos, imprimir ou gerar Excel.

10.2 Saiku

O saiku[17] é uma aplicação *web* que pode ser configurada num servidor ou executada numa máquina de cliente. Sendo executada em modo cliente poderá ser acedida através do endereço : http://localhost:8080/.

Para usar o saiku é necessário obter uma licença no site (http://licensing.meteorite.bi/). A licença do tipo "community " é gratuita. Depois de fazer *download* deverá ser feito upload no URL http://localhost:8080/upload.html

Para ligar ao cubo OLAP Mondrian, é necessário criar uma nova *data source* na área de administração (Admin console > Data Source Management > Add Data Source). Basta atribuir um nome, selecionar a "connection type" como "XMLA" e especificar o URL do cubo: http://<servidor>:8080/mondrian/xmla

Figura 27 - Configuração da data source no Saiku

Estando a fonte de dados configurada, para fazer uma análise exploratória, basta ir à primeira página do Saiku e selecionar “Create new query”. Na lista de cubos seleciona-se o cubo recém-criado e aparece uma lista das medidas e uma lista das dimensões. As dimensões vão ter as hierarquias que foram definidas. Para explorar os dados basta arrastar os campos para as zonas “Medidas”, “Colunas”, “Linhas” ou “Filtros”. Em cada uma destas zonas existe um ícone com uma seta para baixo onde existem opções de ordenação, filtragem, formas de agregação, etc. Para aplicar filtros mais complexos é preciso ter algumas noções de MDX.

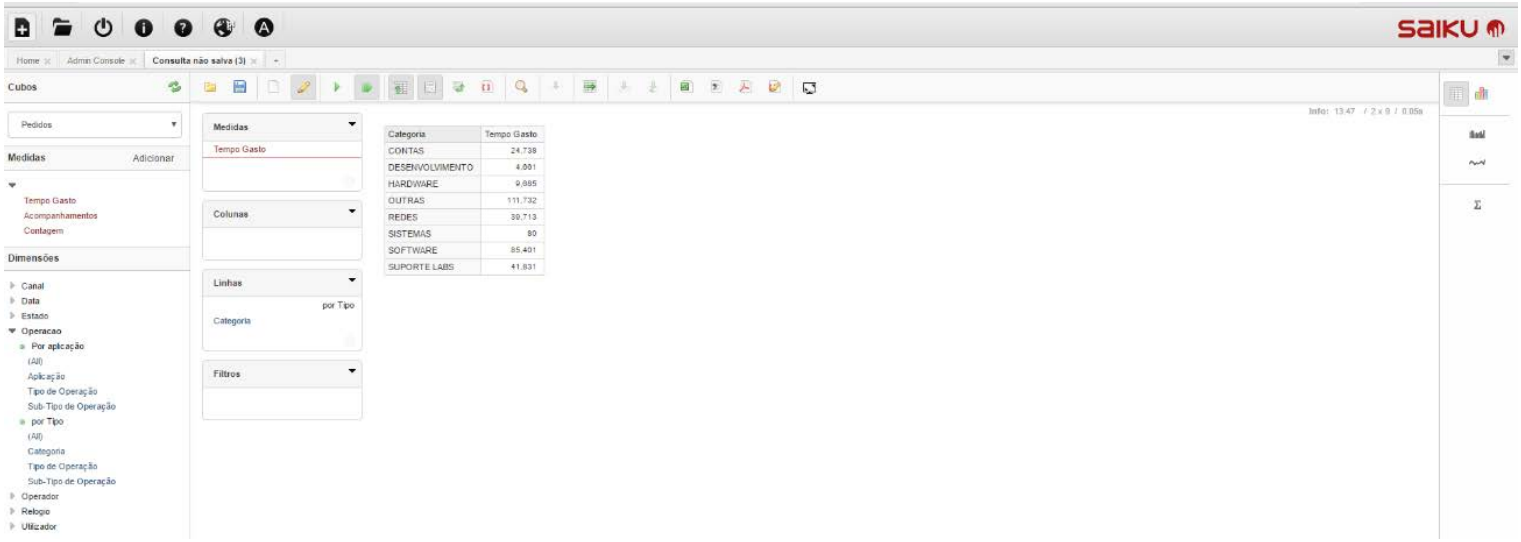


Figura 29 - Resultado em forma de lista no Saiku

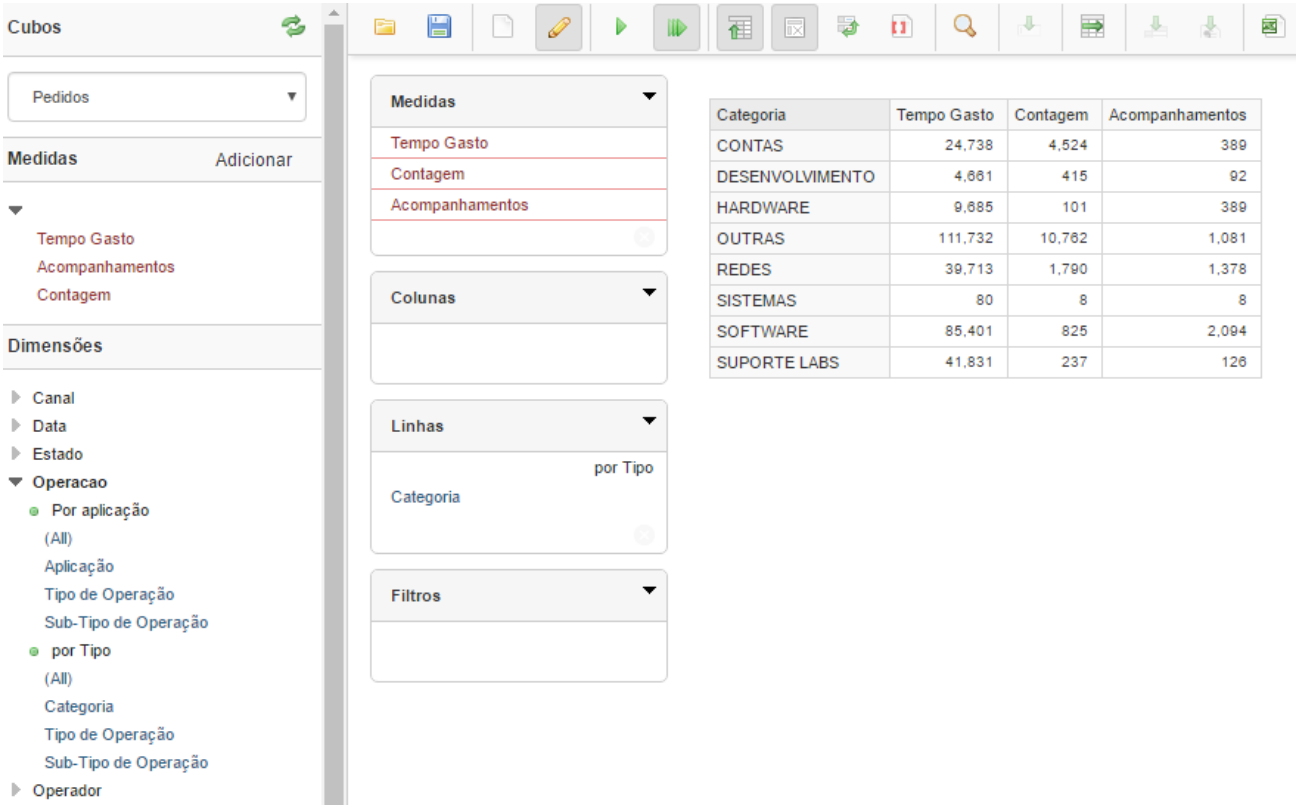


Figura 28 - Resultado em forma de lista no Saiku (detalhe)

O Saiku pode apresentar os resultados em forma de tabela ou em forma de gráficos. Para alternar entre os dois modos ou escolher o tipo de gráfico pretendido existem uns ícones no canto superior direito para esse efeito.

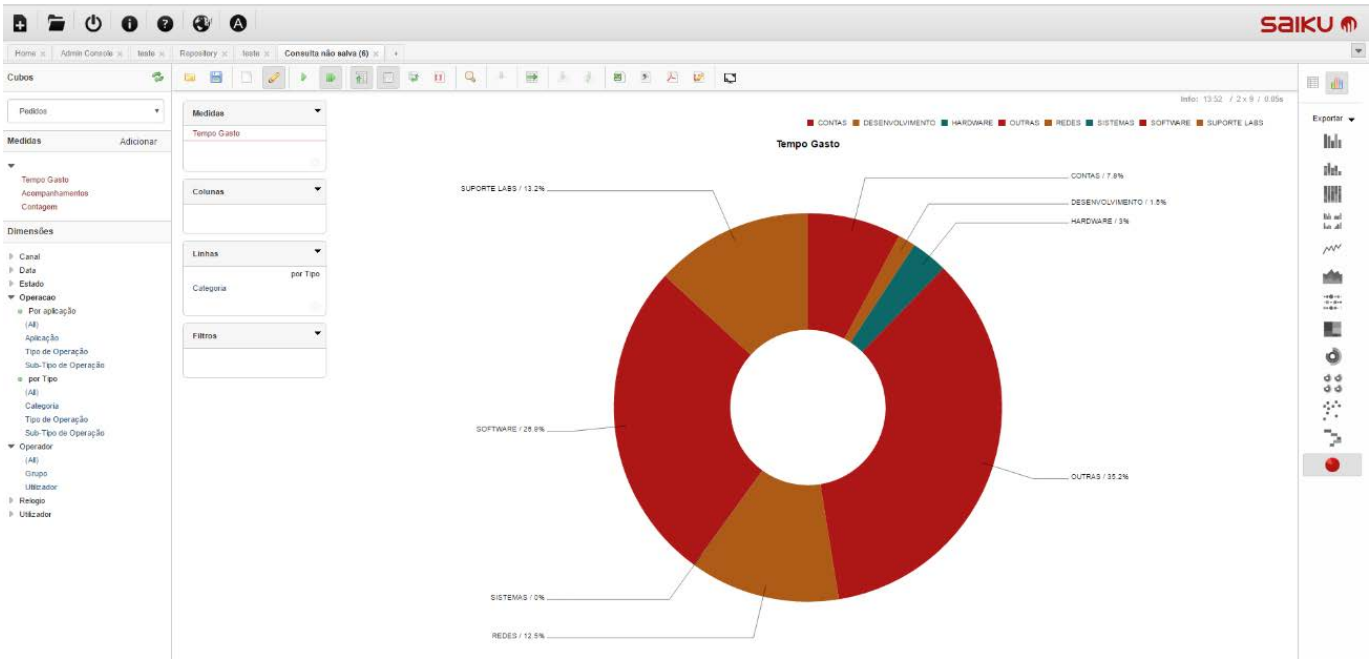


Figura 30 - Resultado em forma de gráfico no Saiku

O saiku também permite exportar dados em PDF e Excel.

Outra funcionalidade importante é a possibilidade de salvar as consultas. Por um lado permite não perder o trabalho efetuado, por outro permite que um perito da área possa usar MDX e fazer *queries* mais complexas que utilizadores normais podem usar.

10.3 *Pivot table excel*

É possível aceder aos dados do cubo OLAP, usando as *pivot tables* do Excel. O Excel tem a grande vantagem de ser uma ferramenta amplamente usada pelos utilizadores e de ter uma interface intuitivo. Para poder usar é necessário instalar o seguinte *software* gratuito: https://sourceforge.net/projects/xmlaconnect/

Quando se insere uma *pivot table*, ao escolher a fonte de dados ”Outros” vai aparecer a opção “XMLA Data Source”. Na opção *location* coloca-se o endereço:

http:// <servidor>:8080/Mondrian/xmla

Apesar de terem sido feitas inúmeras tentativas, não foi possível ligar o *pivot table* ao Excel. Existem vídeos no *youtube* [15] e *blogs* [14] que comprovam que é possível, mas a documentação é escassa e muitas vezes está errada. Pelo o que foi averiguado, só com algumas versões do Mondrian é se consegue efetuar esta ligação.

10.4 Relatórios

Os relatórios são enviados mensalmente via *e-mail*. Existe um *cron* que irá invocar os seguintes *endpoints:*

#	Acção	Endpoints
1	**Público**	\servicoDataWarehouse\TemplatesPDF\Publico
2	**Departamental**	\servicoDataWarehouse\TemplatesPDF\Departamental
3	**Interno**	\servicoDataWarehouse\TemplatesPDF \Interno

Tabela 38 - Lista de endpoints para gerar os reports

Qualquer um destes *endpoints* irá verificar o mês corrente e determinará o intervalo de datas dos dados, gerar o respetivo *template* em PDF, anexá-lo a um email e enviar para o destinatário definido na tabela "reports". A mensagem de email também está definida na base de dados na tabela tipo_report. (Consultar capitulo 7– Desenho Físico)

Ciências ULisboa

Relatório gerado a 2016-05-12 17:42:37

Relatório Público de Serviço da Unidade de Informática

Período: de 2015-03-01 a 2016-05-01

Totalização de Pedidos

Por Natureza

Natureza	Nº de Pedidos	Tempo Médio por pedido	Tempo Total por pedido	Número médio de acompanhamentos	Número Total de acompanhamentos
Pedidos GLPI-Área de Utilizador	201	9 m	1 D 6 h 37 m	1.1990	241
Pedidos GLPI-Configuração Wireless/VPN	48	12 m	10 h 16 m	0.9792	47
Pedidos GLPI-Entrega de Material (indisponível para alunos)	18	9 m	2 h 56 m	1.0000	18
Pedidos GLPI-Hardware	4	28 m	1 h 54 m	1.2500	5
Pedidos GLPI-Impressoras	3	13 m	40 m	1.0000	3

Figura 31 – Exemplo de um relatório público

10.5 Weka

Para o weka[12] poder processar os dados provenientes da DW, estes precisam ser colocados num ficheiro proprietário de extensão .arff que segue uma estrutura especifica. O primeiro passo é extrair dados da DW através de uma *query* de SQL:

```
select c.canal as CANAL_desc,  d.Ano as DATA_ano, d.`Descrição do mês` as DATA_mes,
d.Semestre as  DATA_semestre,  d.`Tipo de dia` as DATA_tipo_dia,
r.`Periodo do dia` as RELOGIO_periodo_dia, o.`Aplicação` as OPERACAO_aplicacao,
o.Categoria as OPERACAO_categoria, o.`Tipo de Operação` as OPERACAO_TIPO,
e.Categoria ESTADO_desc, u.Departamento as USER_Departamento,
 u.`Faixa Etária` as USER_faixa_etaria, u.Grupo as USER_grupo,
u.`Género` as  USER_genero, u.`Tipo de Utilizador` as USER_tipo_utilizador,
 u.Nacionalidade as USER_nacionalidade, f.Acompanhamentos as
MEDIDA_acompanhamentos,  f.`Tempo Gasto` as  MEDIDA_tempoGasto
from fact f inner join  dim_canal  c on  c.ID = f.dim_canal_id
            inner join  dim_data  d on  d.ID = f.dim_data_id
            inner join  dim_relogio  r on  r.ID = f.dim_relogio_id
            inner join  dim_operacao  o on  o.ID = f.dim_operacao_id
            inner join  dim_estado  e on  e.ID = f.dim_estado_id
            inner join dim_utilizador u on u.ID = f.dim_utilizador_id
where u.Nacionalidade <>  'Desconhecido' and u.`Faixa Etária` <> 'Desconhecido
        ' and `Género` <> 'Desconhecido'
```

O segundo passo é exportar o resultado dessa *query*, em que as colunas são separadas por virgulas (“,”) e o delimitador das *strings* é a plica (“ ’ ”).

O terceiro passo é definir um cabeçalho que define as colunas. O weka não funciona bem com *strings* e sempre que seja possível, os atributos devem ser numéricos ou discretizados com uma lista de valores. Os únicos atributos que não são numéricos ou discretizados por terem muitas ocorrências foram o departamento e a nacionalidade. Para ver a estrutura do ficheiro consultar o Anexo C - Ficheiro ARFF para o WEKA.

Para além dos algoritmos de *data mining* que podem ser usados, o WEKA também possui formas de visualização de dados em que se pode combinar vários campos, pode ser também usado como ferramenta analítica.

Capítulo 11 Data Mining

11.1 Enquadramento teórico

Data mining [9] são técnicas computacionais que permitem encontrar padrões e informação "escondida" em dados que os humanos dificilmente conseguiriam encontrar. Estas técnicas dividem-se em dois grandes grupos:

- **Supervisionada:** o processo de aprendizagem é feito através do fornecimento de dados de treino. Estes dados de treino têm intervenção humana. Quando o sistema avaliar novas instâncias o resultado será em função dos dados fornecidos anteriormente. Por exemplo, na implementação de um sistema que classifica se um *email* é *SPAM* ou não. Os dados de treino vão ter *features* como a ocorrência de certas palavras tais como "money", "$", etc. Aplicando algoritmos de classificação, como por exemplo o JRip ou IBK, o sistema vai comparar a classificação que fez com a verdadeira fornecida por humanos. O sistema vai aprendendo e refinando os seus resultados. Quando receber novas instâncias irá prever qual a classe: é *SPAM* ou NOT *SPAM*.
- **Não supervisionada:** O processo de aprendizagem não tem qualquer intervenção humana. Um exemplo são algoritmos de *clustering* em que agrupam instâncias mediante critérios. Nos supermercados estas técnicas são usadas para compreender que produtos os clientes costumam comprar em conjunto. É por esta razão que, por exemplo, as batatas fritas estão perto da cerveja.

Existem várias técnicas como a classificação, os *clustering*, regras de associação, regressão, etc. Independentemente do tipo de técnica a ser aplicada, o processo de *data mining* é composto pelos seguintes passos:

- **Extração dos dados de origem:** processo de ETL combina várias fontes de dados para uma *data warehouse*.
- **Análise dos dados:** O analista de *data mining*, com base no que pretende aferir, analisa os dados e pensa nas melhores técnicas a aplicar, atributos que podem ser descartados, etc.
- **Preparação do dados:** Mediante a análise previamente efetuada e o *software* de *data mining* a ser usado, os dados são trabalhados de maneira a servirem de *input* ao *software*.

- **Análise dos dados pelo software:** O analista aplica o software de *data mining* , testando vários algoritmos, afinando parâmetros até determinar qual é o melhor.
- **Apresentação de resultados:** Os resultados são apresentados em forma de um relatório, gráficos, etc.

À partida não existe um algoritmo / classificador ideal, sendo necessário testá-los a fim de determinar qual o que se adequa melhor aos dados e ao problema. Para tal é preciso comparar alguns indicadores tais como:

Tabela de confusão:

É uma tabela que cruza a classificação que foi prevista com a classificação verdadeira.

Classificados Correctamente	**True Positives (TP)**	**True Negatives (TN)**
Classificados Erradamente	**False Positives (FP)**	**False Negatives (FN)**

Tabela 39 - Tabela de confusão

Após correr um determinado classificador, o Weka devolve os seguintes indicadores:

Indicadores	Descrição
CorrectlyClassifiedInstances	Percentagem de instâncias bem classificadas
IncorrectlyClassifiedInstances	Percentagem de instância que foram mal classificadas
Kappastatistic	É uma relação entre os valores classificados probabilisticamente e os seus valores obtidos pelo classificador. Se for maior que 0 é porque o classificador está a ter algum resultado
Meanabsolute error	Média de erros encontrados
Precision	TP / (TP + FP)
Recall	TP / (TP + FN)
F-Measure	Combina o precision e recall: F = 2 x (precision x Recall) / precision + Recall. Pouco usado por se considerar tendencioso
ROC Área	ReceiverOperatingCharacteristics

Tabela 40 - Lista de indicadores do weka

Os indicadores que vão ser mais usados são ***CorrectlyClassifiedInstances***, ***Precision*** e o ***Recall***

Também é preciso evitar o *overfitting*, ou seja, mesmo que o modelo tenha os melhores indicadores estes podem estar demasiado ajustados aos dados de treino e não são adequados a novos dados.

11.2 Aplicação

Para o contexto deste projeto vão ser utilizadas técnicas de classificação para a previsão do tempo gasto de um pedido. Aplicado a novos pedidos pode servir como valor indicativo do tempo que pode tomar aos operadores ou até mesmo preencher esse tempo gasto automaticamente. Também foram testados algoritmos de *clustering*, mas os resultados não foram o suficientemente interessantes e inesperados para serem referidos neste relatório.

Para poder classificar o tempo gasto de um pedido, foi necessário discretizar a medida "Tempo gasto" em classes de intervalos. O weka tem ferramentas (Filter>supervisioned>attribute>discretize) que passam automaticamente atributos contínuos para conjuntos discretos. Sendo difícil de configurar esse processo automático (é o weka que decide os conjuntos) a fim de produzir as classes pretendidas, foi criado um pequeno programa em JAVA que recebe o ficheiro arff gerado anteriormente e acrescenta um novo atributo do tipo *class (@attributeclass {?,[0-10],[10-30],[30-60],[60-120],[120+]}*), retira a última coluna "MEDIDA_tempoGasto" e substitui pelo respetivo intervalo a que pertence.

Abrindo esse ficheiro, o próximo passo é testar os algoritmos de classificação. A fim de terminar qual o tipo de algoritmo que se adapta melhor ao problema, foram usados os seguintes:

- **Árvores de decisão**
  - J48
  - RandomTree,
  - RandomForest
- **Regras de classificação**
  - JRip
  - PART
- **Aprendizagem baseada em instâncias**
  - IBK
  - KStar
- **Redes bayesianas**
  - BayesNet
- **Modelo estatístico**
  - NaiveBayes

Na experimentação dos algoritmos foi usada a opção *percentage split* em que 66% é usada para treino e 33% para teste. Os primeiros resultados foram (para ver todo o *output* consultar o Anexo D - Resultados dos algoritmos de classificação do weka) :

Algoritmo	***Correctly Classified Instances***	***Precision* (media)**	***Recall* (media)**
J48	82.8444 %	*0,837*	*0,828*
RandomTree	*77.9111 %*	*0,776*	*0,779*
RandomForest	*81.9111 %*	*0,818*	*0,819*
JRip	*81.0222 %*	*0,827*	*0,810*
PART	*81.4222 %*	*0,822*	*0,814*
IBK	*78.7556 %*	*0,782*	*0,788*
KStar	*79.8667 %*	*0,796*	*0,799*
BayesNet	*72.3556 %*	*0,776*	*0,724*
NaiveBayes	*74.3556 %*	*0,773*	*0,744*

Tabela 41 - Resultados a aplicação dos algoritmos weka.

Os resultados preliminares aplicados aos dados de teste andam por volta dos 72-82 %. Os algoritmos mais bem classificados são: J48 (82.8 %), RandomForest (*81.9 %*), PART (*81.4 %).* Tendo encontrado os melhores candidatos, o próximo passo é alterar os parâmetros com a finalidade de conseguir ainda a melhores resultados.

No J48, alterando o parâmetro “ConfidenceFactor” de 0.25 para 0.35, o número de instâncias corretamente classificadas passou de 82.8 % para 83.2 %.

No RandomForest, consegue-se passar de 100 para 110 instâncias com melhorias, passando o número de instâncias classificadas corretamente de *81.9 %* para 82.2%. Se se utilizar mais do que 110 instâncias, o algoritmo fica demasiado e lento para ser praticável usá-lo.

No PART, não houve afinações que tivessem um impacto suficientemente grande para serem mencionados.

Para obter melhores resultados e mitigar o problema do *overfitting*, estes algoritmos podem ser combinados. Uma das melhores formas é usar a votação em que a classificação é votada entre os resultados dos três algoritmos. Testando as várias regras de combinação, determinou-se que o "*majority voting*" apresentou melhores resultados com 82.7 % instâncias bem classificadas. Apesar de ser um pouco menor que o J48 com 83.2 %, a solução de combinar três algoritmos garante que o modelo vai avaliar melhor novos dados.

Encontrados os melhores algoritmos, o próximo passo seria a implementação que passaria pela construção de um *servlet* que recebia os dados de um novo pedido, usava o modelo aprendido na fase de treino e devolvia o tempo estimado. Este *servlet* poderia ser integrado com as aplicações de modo a que o tempo devolvido fosse meramente indicativo para o operador ou até mesmo automaticamente esse valor.

Capítulo 12 Resultados e respostas às perguntas analíticas

Vão ser respondidas as dezassete perguntas analíticas que foram formuladas no "capítulo 5 - Processos de negócio" referentes ao *data mart* dos pedidos, visto ser o único que foi implementado.

12.1 Qual o tipo de operações que despendem mais tempo?

Para responder a esta questão foram acrescentadas as medidas "Tempo Gasto" (Alterou-se a *query* de MDX para devolver no formato Horas Minutos) e nas linhas o campo "Categoria" da dimensão Operação. Ordenou-se a lista por tempo gasto descendente. Foram obtidos os seguintes resultados:

Categoria	Tempo Gasto
OUTRAS	1862 h 12 m
SOFTWARE	1423 h 21 m
SUPORTE LABS	697 h 11 m
REDES	661 h 53 m
CONTAS	412 h 18 m
HARDWARE	161 h 25 m
DESENVOLVIMENTO	77 h 41 m
SISTEMAS	1 h 20 m

Tabela 42 - Resultados de tempo gasto por categoria de operação

Conforme se pode verificar, as operações que consomem mais tempo são aquelas cuja categoria não se consegue encaixar em nenhuma. Este resultado pode indiciar que seja necessário repensar e restruturar a categorização dos tipos de operação. Em segundo lugar vem a categoria de SOFTWARE. Fazendo um *drill-down* em SOFTWARE verifica-se que o tempo consumido prende-se sobretudo com a "instalação do sistema operativo" e "instalação de outro software".

SOFTWARE	1423 h 21 m
Instalação de Sistema Operativo	756 h 19 m
Instalação de Sistema Operativo (indisponível para alunos)	57 h 2 m
Instalação de Software	452 h 50 m
Instalação de Software (indisponível para alunos)	18 h 8 m
Outros Problemas	138 h 57 m
Vírus	0 h 5 m

Tabela 43 - Drill-down da categoria de Operação "SOFTWARE"

Quando os computadores são novos ou foram formatados, procede-se à instalação do sistema operativo. Apesar de esta operação consumir muito tempo, felizmente necessita pouca intervenção humana, libertados os operadores para outras tarefas, enquanto instala o *software*.

12.2 Quais são as operações mais frequentes?

Para responder a esta questão foi usada a medida “Contagem”, como colunas o campo “Aplicação” e nas linhas o campo “Categoria”. Ambas da dimensão Operação. Foram obtidos os seguintes resultados:

	Gestão Grupos	Inbox	Pedidos	Pedidos GLPI	Tomadas	
Categoria						**Total**
OUTRAS		6,802	3,608	352		**10.762**
CONTAS			4,281	243		**4.524**
REDES		1	525	57	1,207	**1.790**
SOFTWARE			785	40		**825**
DESENVOLVIMENTO			400	15		**415**
SUPORTE LABS			226	11		**237**
HARDWARE			68	33		**101**
SISTEMAS	8					**8**

Categoria	Gestão Grupos	Inbox	Pedidos	Pedidos GLPI	Tomadas	Total
Total	8	6.803	9.893	751	1.207	

Tabela 44 – Resultado com a contagem de pedidos entre a categoria de operação e Aplicações

Combinando o tipo de operação com as aplicações utilizadas, pode-se verificar que as operações mais comuns provêem das aplicações de pedidos e em operações não especificadas. Como foi verificado na questão anterior, existem muitos pedidos de *software*.

12.3 Qual a distribuição de operações ao longo de um ano letivo / semestre?

Para responder a esta questão foi utilizada a medida “Contagem”, como coluna o campo “Categoria” da dimensão Operação e como linhas os campos “Ano”, “Semestre” e “Mês” da dimensão Data.

Mês	CONTAS	DEV	HARDWARE	OUTRAS	REDES	SISTEMAS	SOFTWARE	SUPORTE LABS	Total
1	306	40	4	674	130	2	72	18	1.246
2	280	31	9	769	109		80	19	1.298
3	288	25	6	794	157		85	26	1.381
4	254	30	5	579	106		77	21	1.072
5	173	33	5	671	126		63	26	1.097
6	145	37	6	649	89		65	19	1.010
7	129	29	4	806	194		73	21	1.256
8	109	7	1	444	35		25	15	636
9	1,026	92	7	2,572	206		77	26	4.006
10	676	39	8	985	162	5	67	14	1.956
11	609	22	3	892	160		64	10	1.760
12	286	15	10	575	119		37	11	1.053

Tabela 45 - Distribuição da contagem de pedidos por categoria de Operação ao longo do tempo

O pico é no mês de setembro devido ao início do ano letivo. O mês de Agosto é o mês com menos atividade porque é período de férias. Os meses com uma atividade acima da média são Outubro e Novembro. A explicação é por um lado a entrada de novos alunos na segunda fase que explica o valor acentuado de pedidos relacionados com contas, por outro ainda são resquícios de operações de início de ano letivo.

12.4 Qual a distribuição do tempo gasto em relação aos departamentos?

Para responder a esta questão foi utilizada a medida "tempo gasto", nas linhas o campo "Departamento" da dimensão utilizador ordenada por "tempo gasto" descendente e com a opção TOP 10. Foram obtidos os seguintes resultados:

Departamento	Tempo
Centro de Informática	1384 h 17 m
Departamento de Biologia Animal	414 h 50 m
Departamento de Matemática	341 h 0 m
Departamento de Química e Bioquimica	286 h 57 m
Departamento de Engenharia Geografica Geofisica e Energia	228 h 16 m
Departamento de Física	220 h 8 m
Depart. de Estatística e Invest. Operac.	158 h 47 m
Departamento de Geologia	158 h 31 m
Departamento de Biologia Vegetal	144 h 58 m
Instituto de Oceanografia	142 h 25 m

Tabela 46 - Listagem de tempo gasto por departamento

O número de tempo gasto pelo "Centro de informática" deve-se ao fato de muitas vezes se fazer pedidos internos com o utilizador "Suporte Informático (FC)". Esta ação pode ser benéfica por colocar no sistema operações que foram realizadas, mas por outro é informação que se perde, porque, o tempo gasto destes pedidos foi certamente para alguém externo ao "Centro de Informática".

Em relação aos outros departamentos e combinando com o tipo de operação:

	CONTAS	DESENVOLVIMENTO	HARDWARE	OUTRAS	REDES	SOFTWARE	SUPORTE LABS
Departamento							
Centro de Informática	510	11	296	39,593	1,980	5,002	35,655
Departamento de Biologia Animal	3,811	205	579	7,126	1,998	7,770	3,401
Departamento de Matemática	1,780	286	2,116	4,876	4,264	7,138	
Departamento de	2,496	365	50	6,314	2,645	4,275	1,042

	CONTAS	DESENVOLVIMENTO	HARDWARE	OUTRAS	REDES	SOFTWARE	SUPORTE LABS
Departamento							
Química e Bioquimica							
Departamento de Engenharia Geografica Geofisica e Energia	1,906	128	389	2,076	3,460	5,506	231
Departamento de Física	2,061	82	153	2,308	2,097	6,112	395
Departamento de Geologia	2,010	55	55	2,080	997	4,125	189
Departamento de Informática	3,222	237	12	1,235	879	104	

Tabela 47 - Departamentos cruzando com categoria de operação

Verifica-se o Departamento de Biologia e de Química despendem tempo no suporte a laboratórios, ao passo que o departamento de Matemática despende muito tempo em pedidos relacionados com hardware. De salientar que o tempo gasto no Departamento de Informática ser mais baixo que nos outros departamentos porque este tem alguma autonomia e efetuam muitas das tarefas internamente.

12.5 Existem muitos pedidos pendentes?

Para responder a esta questão foi utilizada a medida “Contagem” e como linhas o campo “Estado genérico” da dimensão Estado. Foram obtidos os seguintes resultados:

Estado genérico	Contagem
CONCLUIDO	12,285
OUTRO	6,029
NOVO	326
EM CURSO	13
PENDENTE	9

Tabela 48 - Contagem de pedidos distribuído por estado

Como era de esperar, a maioria dos pedidos estão concluídos devido às políticas da UI de resolver os pedidos em menos de 48 horas. Relativamente ao Estado “OUTRO” refere-se ao estado “read” da aplicação “Inbox”. Um *email* que foi lido não implica que

se tenha materializado numa tarefa ou que esteja concluído. Por essa razão deverá ser ignorado. O mesmo para o caso do estado “NOVO”. Três pertencem à aplicação Pedidos GLPI e correspondem a últimos pedidos da última extração de dados. Relativamente aos outros 323, pertencem ao estado “unread”. Um *email* não lido significa que não foi tratado. sendo *emails* que pelo remetente e assunto os operadores verificaram que era SPAM, mas nunca chegaram a arquivar ou apagar.

12.6 Que tipo de utilizador despende mais recursos? Alunos, Funcionários ou Docentes?

Para responder a esta questão foram usadas as medidas “Tempo gasto”, “Acompanhamentos” e “Contagem”. Como linhas o campo “Tipo” da dimensão utilizador. Foram obtidos os seguintes resultados:

Tipo	Tempo Gasto	Acompanhamentos	Contagem
FUNCIONARIO	193,768	2,674	6,970
DOCENTE	96,026	2,575	3,808
ALUNO	28,045	306	4,914

Tabela 49 - Distribuição da contagem, tempo gasto e acompanhamentos por tipo de utilizador

Os funcionários são os que despendem mais tempo. Apesar de os Docentes terem quase metade dos pedidos dos funcionários, têm aproximadamente o mesmo número de acompanhamentos. Os alunos apesar de terem mais pedidos que os docentes (por serem em maior número) despendem bastante menos tempo.

12.7 Qual o canal que consome mais recursos?

Para responder a esta questão foram usadas as medidas “Tempo gasto”, “Acompanhamentos” e “contagem”. Como linhas foi usada o campo “Canal” da dimensão “Canal”. Foram obtidos os seguintes resultados:

Canal	Tempo Gasto	Acompanhamentos	Contagem
Telefone	192,633	1,761	5,277
Presencial	64,929	2,742	3,921
web	43,198	960	8,746
E-mail	17,081	94	718

Tabela 50 - Tempo gasto, Acompanhamentos e Contagem distribuídos por canal

Como se pode verificar o canal via telefone é o que consome mais tempo. Por outro lado o canal *web* apesar de ter mais pedidos, comparativamente tem menos tempo gasto.

12.8 Existe alguma correlação entre os recursos despendidos e a faixa etária dos utilizadores?

Para responder a esta questão foram usadas as médias "Tempo Gasto", "Contagem" e "Acompanhamentos". Foram criadas duas medidas adicionais calculadas: rácio Tempo Gasto / Contagem e rácio Tempo Gasto / Acompanhamentos. Nas linhas o campo "Faixa Etária" da dimensão utilizador. Foram obtidos os seguintes resultados:

Faixa Etária	Tempo Gasto	Contagem	Racio Tempo / Contagem	Acompanhamentos	Racio Tempo / Acompanhamentos
[18-20]	3,169	551	5.751	38	83.395
[21-23]	2,300	464	4.957	34	67.647
[24-26]	2,418	308	7.851	59	40.983
[27-30]	5,573	416	13.397	204	27.319
[31-40]	27,066	1,446	18.718	820	33.007
[41-50]	34,386	1,280	26.864	831	41.379
[51 - 65]	53,828	1,898	28.36	1,267	42.485
[65+]	5,597	261	21.444	156	35.878

Tabela 51 - Distribuição de medidas por faixa etária

Pode-se constatar que quanto maior for a idade, maior é o tempo gasto. O intervalo de idade que consome mais recursos situa-se entre os 40 e os 65 anos. Associando a faixa etária com o tipo de utilizador:

	ALUNO	DOCENTE	FUNCIONARIO
Faixa Etária			
[65+]		4,729	868
[51 - 65]	190	40,211	13,427
[41-50]	220	13,157	21,009
[31-40]	820	7,515	18,731
[27-30]	533	285	4,755
[24-26]	923	84	1,411
[21-23]	2,297		3
[18-20]	3,169		

Tabela 52 - Distribuição de faixa etária por tipo de utilizador

Pode-se depreender que os alunos no início recorrerem mais aos serviços da UI, mas à medida que vão passando de ano e envelhecendo vão recorrendo cada vez menos.

Por outro lado, os docentes mais velhos consomem bastante tempo, sobretudo os da faixa etária [51-65]. De notar que os docentes e funcionários dos outros escalões têm valores bastante aproximados o que se pode depreender alguma dificuldade de adaptação às novas tecnologias.

12.9 Existe alguma correlação entre o tempo gasto e o género?

Para responder a esta questão foram usadas as medidas "Tempo Gasto", "Acompanhamentos" e "Contagem". Como colunas foi usado o campo "Género" e como linha o campo "Tipo", ambas da dimensão Utilizador. Foram obtidos os seguintes resultados:

	Feminino			Masculino		
Tipo	**Tempo Gasto**	**Acompanhamentos**	**Contagem**	**Tempo Gasto**	**Acompanhamentos**	**Contagem**
ALUNO	12,779	122	2,030	13,104	135	2,552
DOCENTE	42,124	1,042	1,640	51,696	1,441	2,068
FUNCIONARIO	55,669	1,275	2,246	36,527	861	1,332

Tabela 53 - Medidas cruzando o tipo de utilizador com o seu género

Em relação aos alunos e docentes, os valores são bastante equilibrados entre os dois géneros. Em relação aos funcionários, as mulheres consomem um pouco mais de tempo que os homens já que o rácio nos funcionários é de 1,29 de mulheres em relação a homens e no tempo gasto é de 1,55.

12.10 Existe uma correlação entre o número de acompanhamentos e o tempo gasto?

Para responder a esta questão foram usadas as medias "Tempo Gasto" e "Acompanhamentos". Foi criada a medida rácio Tempo Gasto / Acompanhamentos . Nas linhas foi usado o campo "Aplicação " da dimensão Operação. Foram obtidos os seguintes resultados:

Aplicação	**Tempo Gasto**	**Acompanhamentos**	**Racio Tempo / Acompanhamentos**
Gestao Grupos	80	8	10
Pedidos	256,071	3,357	76.28
Pedidos GLPI	27,535	985	27.954

Aplicação	Tempo Gasto	Acompanhamentos	Racio Tempo / Acompanhamentos
Tomadas	34,155	1,207	28.297

Tabela 54 - Distribuição das medidas por tipo de aplicação

Listando as medidas “Tempo Gasto”, “Acompanhamentos” e a medida calculada “Racio Tempo /Acompanhamentos” podemos aferir que na antiga aplicação de pedidos cada acompanhamento durava bastante mais tempo que na nova aplicação de pedidos ou na de tomadas. A razão pode-se prender com a forma como os operadores colocam os tempos despendidos em cada tarefa. Na antiga aplicação de pedidos esse tempo era introduzida manualmente e provavelmente era muitas vezes inflacionado. Nas novas aplicações esse tempo ou é atribuído automaticamente ou em intervalos discretos.

12.11 Quais os tipos de operação com mais acompanhamentos?

Para responder a esta questão foi usada a medida “Acompanhamentos”. Como coluna foi utilizado o campo “Aplicação” e como linhas os campos “Categoria” e “Tipo de operação”. Todos da dimensão Operação. Foram obtidos os seguintes resultados:

	Gestao Grupos	Pedidos	Pedidos GLPI	Tomadas	
Categoria					**Total**
CONTAS		105	284		**389**
DESENVOLVIMENTO		76	16		**92**
HARDWARE		345	44		**389**
OUTRAS		671	410		**1.081**
REDES		113	58	1,207	**1.378**
SISTEMAS	8				**8**
SOFTWARE		1,964	130		**2.094**
SUPORTE LABS		83	43		**126**

Tabela 55 - Distribuição dos acompanhamentos entre categorias de operação e aplicações

Como se pode verificar nesta tabela, o tipo de operação com mais acompanhamentos é a categoria “SOFTWARE”. A razão para tal acontecer deve-se ao facto de quando se prepara uma máquina para disponibilizar a um utilizador, depois de esta ser formatada, são instalados o sistema operativo e *software* diverso. Estas ações são realizadas por vários operadores, dai a razão de um número tão elevado.

Logo a seguir são as tomadas de rede, porque todos os pedidos no site de tomadas implica pelo menos um acompanhamento do técnico de redes.

E em terceiro lugar, a categoria “Outras” provenientes das duas aplicações de pedidos.

12.12 Quais as nacionalidades dos utilizadores não portugueses mais comuns e os recursos que consomem?

Para responder a esta questão foram usadas as medidas “Contagem”, “Tempo Gasto”, “Acompanhamentos”:

Nacionalidade	Contagem	Tempo Gasto	Racio Tempo / Contagem	Acompanhamentos	Racio Tempo / Acompanhamentos
Portugal	9,727	185,522	19.073	4,342	42.727
Bélgica	72	1,299	18.042	32	40.594
Brasil	56	478	8.536	0	0
Cabo Verde	56	248	4.429	3	82.667
Angola	53	313	5.906	4	78.25
Itália	32	152	4.75	3	50.667
Guiné Bissau	29	104	3.586	0	0
Alemanha	27	182	6.741	11	16.545
Ucrânia	23	70	3.043	6	11.667

Tabela 56 - Lista de nacionalidades e suas respectivas medidas

Portugal como era esperado tem um número de pedidos esmagador em relação às outras nacionalidades, mas se olharmos para os rácios podemos verificar que a Bélgica tem valores semelhantes aos portugueses. A grande maioria desses pedidos são efetuadas por um docente de nacionalidade belga. As outras nacionalidades vão diminuindo drasticamente o número de pedidos e os seus rácios. Discriminando por tipo de Utilizador temos:

	ALUNO	DOCENTE	FUNCIONARIO
Nacionalidade			
Portugal	3,408	3,359	2,960

	ALUNO	DOCENTE	FUNCIONARIO
Nacionalidade			
Bélgica		72	
Brasil	50	2	4
Cabo Verde	56		
Angola	53		
Itália	11	16	5
Guiné Bissau	29		
Alemanha	12		15
Ucrânia	23		

Tabela 57 - Lista de nacionalidades cruzando com tipo de utilizador

Como se pode verificar a maioria dos estrageiros são alunos. Verifica-se que nos lugares cimeiros estão países de língua portuguesa. Tendo todos os alunos à partida as mesmas necessidades, pode-se aferir que a língua poderá ser uma barreira. Alunos que não falem português podem sentir-se dissuadidos de recorrer aos serviços da UI sejam eles web (em que grande parte dos conteúdos só estão em português) ou presencialmente.

12.13 Qual a relação entre o número de pedidos de utilizadores externos e internos?

Para responder a esta questão foi utilizada a medida “Contagem”. Como coluna foi usado o campo “Categoria” da dimensão Operação e como linha o campo “Departamento” da dimensão Utilizador. Foram obtidos os seguintes resultados:

Estatísticas	Contagem
Mínimo	1.000
Máximo	2970.000
Soma	18471.000
Média	167.918
Desvio Padrão	411.890

Tabela 58 - Dados estatísticos da lista de utilizadores

Consultando a lista de Utilizadores, os externos são 2970. Ativando o modo estatístico do Saiku, aferiu-se que o número de pedidos no sistema são 18471. Os

pedidos externos são 16 % do total. É mais do dobro do Departamento de Biologia Animal que é o departamento com mais pedidos (7%).

12.14 Existe alguma correlação entre um tipo de operação e um canal preferencial?

Para responder a esta questão foi utilizada a medida “Contagem”. Como coluna o campo “Canal” da dimensão Canal e como linha o campo “Categoria” da dimensão Operação. Foram obtidos os seguintes resultados:

	Telefone	Presencial	Web	E-mail
Categoria				
OUTRAS	2,081	105	7,939	637
SOFTWARE	550	222	46	7
SUPORTE LABS	200	9	27	1
REDES	474	1,271	41	4
CONTAS	1,875	2,244	361	44
HARDWARE	36	32	31	2
DESENVOLVIMENTO	61	30	301	23
SISTEMAS		8		

Tabela 59 - Distribuição de canal por categoria de operação

A razão para o valor da categoria “Outras” em relação ao canal *web* ser tão elevado deve-se ao facto da aplicação “Inbox” gerar muitos pedidos. Retirando o caso do desenvolvimento em que o canal web tem vantagem, nas outras categorias é pouco significativo. Em relação às “CONTAS” é natural que o canal mais usado seja o “presencial”, porque se os utilizadores não têm conta, não podem aceder às aplicações web.

12.15 Qual é efetivamente o uso das aplicações nos fins-de-semana e feriados?

Para responder a esta questão foi utilizada a medida “Contagem”. Como coluna foi usado o campo “Aplicação” da dimensão utilizador e como linhas os campos ”Tipo de dia” e “Dia da Semana” da dimensão Data. Foram obtidos os seguintes resultados:

		Gestao Grupos	Inbox	Pedidos	Pedidos GLPI	Tomadas
Tipo de dia	**Dia da semana**					
Dia de semana	**3ª feira**		1,348	2,329	190	254
	2ª feira	3	1,318	2,266	136	281
	4ª feira	4	1,317	1,875	159	235
	5ª feira		1,244	1,792	159	223
	6ª feira	1	1,265	1,593	105	204
Fim de semana	**Sabado**		142	7	2	7
	Domingo		127	5		2
Feriado	**5ª feira**		27	23		1
	Sabado		6	1		
	4ª feira		6			
	6ª feira		2	1		
	Domingo		1	1		

Tabela 60 - Uso das aplicações ao longo dos dias de semana

Como era previsível nos fins de semana e feriados, existe uma quebra drástica nos pedidos. Os feriados ainda têm quebras maiores.

O dia com mais pedidos costuma ser as 3ª Ferias e o dia com menos as 6ª Ferias.

12.16 Qual o período do dia com mais atividade? E com menos?

Para responder a esta questão foi usada a medida “Contagem”. Como coluna foi usado o campo “Categoria” da dimensão Operação e como linha os campos “Periodo” e “Hora” da dimensão Relógio. Foram obtidos os seguintes resultados:

		CONTAS	DEV	HARDWARE	OUTRAS	REDES	SOFTWARE	SUPORTE LABS
Periodo	**H**							
Manhã	**7**	5			36	2		
	8	197	12		213	33	20	3
	9	309	35	9	584	83	42	3
	10	588	60	8	1,044	202	89	5

		CONTAS	DEV	HARDWARE	OUTRAS	REDES	SOFTWARE	SUPORTE LABS
Periodo	**H**							
	11	554	63	14	1,333	214	131	18
	12	488	28	7	906	181	62	9
	13	411	12	6	694	157	56	3
Tarde	**14**	456	46	10	1,079	206	108	3
	15	481	60	15	1,121	213	75	8
	16	419	38	14	1,015	212	81	14
	17	295	34	6	924	123	77	22
	18	150	17	7	626	74	34	42
	19	101	2	3	442	49	35	41
Noite	**20**	62	1	1	262	28	3	50
	21	4	3	1	108	3	2	10
	22		3		73	5	8	5
	23		1		79	3		1
	0				46	2	2	
	1				25			
	2	4			113			
	3				7			
	4				8			
	5				10			
	6				14			

Tabela 61 - Distribuição das categorias de operação ao longo do dia

Os picos de trabalho situam-se entre as 10:00 e as 11:00 horas da parte da manhã e das 14:00 às 16:00. Os pedidos a partir das 22:00 horas baixam significativamente retornando novamente a atividade a partir das 8:00 horas.

Os pedidos de suporte a laboratórios são feitos geralmente no final do dia, provavelmente para o dia seguinte.

12.17 Existem diferenças de tempo gasto entre Técnicos e operadores?

Para responder a esta questão foi usada a medida “Tempo Gasto”. Como coluna foi usado o campo “Ano” da dimensão Data e como linha o campo “Grupo” da dimensão operador. Foram obtidos os seguintes resultados:

	2015	2016
Grupo		
Operador	20 h 0 m	877 h 30 m
Técnico	28 h 0 m	1228 h 30 m

Tabela 62 - Diferença de tempo gasto entre técnicos e operadores

O registo efetivo do tempo gasto dos técnicos e operadores só pode ser efetivamente feito na aplicação do site das tomadas e na nova aplicação Pedidos GLPI. Por esse motivo é que os cinco meses de 2016 apresentam muito mais horas que 2015. Cingindo aos dados de 2016 e acrescentando as medidas “contagem” e a medida calculada do “rácio entre tempo e contagem”, temos:

	2016		
Grupo	**Tempo Gasto**	**Contagem**	**Rácio Tempo / Contagem**
Operador	52,650	2,100	25.071
Técnico	73,710	2,940	25.071

Tabela 63 - Tempo gasto e contagem por grupo em 2016

Pode aferir-se que sendo o rácio o mesmo, quer para operadores, quer para técnicos de tempo gasto / contagem ser de 25 minutos por pedido, pode aferir-se que os técnicos consomem mais tempo que os operadores pelo simples facto de terem mais pedidos associados. A questão do rácio ser igual pode significar que a atribuição de tempo gasto na nova aplicação GLPI é constante.

Conclusão

Trabalho realizado

A realização deste projeto provou ser possível construir uma *data warehouse* exclusivamente a partir de ferramentas gratuitas *open source*. O intuito deste relatório é o de ser uma ferramenta que permita a qualquer instituição, mesmo que pequena, ter a possibilidade de construir um sistema de apoio à decisão usando poucos recursos e sem precisar de especialistas da área. Também demonstrou mais uma vez que a aplicação de boas práticas, nomeadamente a metodologia de Kimball, permitiram que a construção fosse feita de forma solida, segura e escalável.

Neste processo com avanços e retrocessos, altos e baixos, a maioria dos objetivos propostos foram superados. Na fase inicial de levantamento de requisitos houve alguma dificuldade em explicar e demonstrar o que era um sistema DW / BI e que podia ter utilidade no dia-a-dia para quem o usa. Através de protótipos e provas de conceito (sobretudo com o JPivot) os utilizadores reconheceram potencial em fazer exploração de dados num sistema OLAP .

Na fase de modelação dimensional não foram encontradas grandes dificuldades, visto serem *data marts* relativamente simples. A única exceção foi o *data mart* de redes, devido à complexidade em analisar as suas ferramentas de monitorização.

Na fase de arquitetura e escolha de produtos, a fase mais crítica foi analisar as dezenas de soluções que existem no mercado. São todas muito similares, perdendo-se muito tempo em comprovar se são compatíveis entre si.

Na construção do ETL, logo desde cedo se percebeu que a abordagem *batch* não servia devido à enorme quantidade de passos com dependências entre si. A solução de utilizar a ferramenta *Kettle*, como *data flow*, invocando serviços em PHP, provou ser elegante, porque garante a separação entre o fluxo do processo por um lado e por outro centraliza todo o código num único projeto

com uma linguagem de programação (PHP) que praticamente não tem limitações.

A escolha do Mondrian foi acertada por ser um produto rápido e estável. No entanto a documentação está desatualizada e a construção do *schema* em XML mesmo com o auxílio do *schema workbench* pode ser penosa. O MDX é uma mais-valia porque permite efetuar exploração de dados mais complexos. No entanto é uma linguagem que requer alguma aprendizagem e provavelmente não é acessível a um utilizador comum. O XMLA permite que o Mondrian possa ligar-se a muitas outras ferramentas.

Em relação às ferramentas analíticas, o JPivot, apesar de simples, provou ser decisiva na construção do cubo de dados em Mondrian quer na sua configuração, quer na construção do *schema*. O Saiku é uma ferramenta de utilização agradável e que integra facilmente com o Mondrian através de XMLA. É uma ferramenta que tanto pode correr localmente na máquina do utilizador como ser instalada num servidor. As perguntas analíticas foram na sua maioria respondidas somente com *drag and drop* de medidas e atributos das dimensões. Para questões mais complicadas, a edição da *query* é uma vantagem. Infelizmente não foi possível ligar as *pivot tables* do Excel ao *data warehouse*. Apesar de haver alguns *blogs* e vídeos do *youtube* que demonstram ser possível, a documentação é escassa e muitas vezes errada. O Excel, ferramenta que todos os utilizadores usam teria sido uma mais valia.

O *weka* demonstrou ser uma boa ferramenta para explorar técnicas de *data mining*. A previsão do tempo gasto de um novo pedido apresentou um resultado de cerca de 82 % de instâncias bem classificadas, sendo o suficiente para ser implementada.

Objetivos pessoais

Este projeto ajudou-me a consolidar os conhecimentos que tinha aprendido academicamente e permitiu-me adquirir conhecimentos para implementar uma *data warehouse* com ferramentas *open source*. Uma das experiências mais relevantes prendeu-se com o facto de esta solução ter sido desenvolvida a partir de um caso real. A interação com os futuros utilizadores, sistemas operacionais diversos e cheios de não conformidades e a escolha de um conjunto de produtos permitiu-me compreender que os livros da especialidade não conseguem prever todas as situações que podem

acontecer. No entanto, a metodologia foi fundamental para o sucesso do projeto e que sem ela a solução teria menos qualidade e demorado mais tempo.

Com a elaboração deste relatório aprendi boas práticas na estruturação e formatação de documentos. Foi desafiante compilar e estruturar todas as ideias relativas a um projeto que tem muitas fases e muito a dizer de uma forma que por um lado não seja demasiado técnica e que por outro não seja demasiado teórica.

Comparação com outras ferramentas

As ferramentas Microsoft SSIS e SSAS comparativamente com a solução apresentada continuam a ser superiores. No entanto, o trabalho desenvolvido demonstra que com um custo reduzido é possível criar algo que se aproxima bastante e que tem menos restrições tecnológicas.

Algumas das tecnologias usadas neste projeto possuem versões superiores que são pagas, como por exemplo, o Saiku que tem uma versão *enterprise* e o Mondrian é mais evoluído na versão integrada no Pentaho.

Hipoteticamente, o único tipo de ferramentas que poderia valer a pena investir seriam as analíticas. Existem soluções no mercado muito mais poderosas no que toca à apresentação dos dados, possibilidade de serem visualizados em dispositivos móveis, etc. O Mondrian permite que estas ferramentas comuniquem com todo o resto da DW implementada gratuitamente.

Análise dos resultados

Ao fazer uma análise exploratória aos dados produzidos no *data mart* dos pedidos, deu para perceber o trabalho meritório que a UI fornece aos alunos, docentes e funcionários da FCUL. Para além de serem disponibilizadas uma gama vasta de serviços a um número vasto de utilizadores, a UI consegue resolver os pedidos em tempo útil.

No entanto existem aspetos que podiam ser melhorados, nomeadamente repensar a categorização das operações. Existem muitos pedidos com operações indiferenciadas com categorias “Outras” em que se perde onde realmente se gastou o tempo. Possivelmente os operadores escolhem a categoria “Outros” porque têm dificuldade em encontrar na lista disponível uma operação que encaixe na ação efetuada.

Outro problema semelhante é o facto de o departamento / unidade da FCUL que tem mais pedidos ser a própria UI. Isto deve-se ao facto de muitas vezes os pedidos

serem feitos com contas internas da UI para registar ações que foram feitas para outras pessoas. Mais uma vez existe perda de informação.

Outra questão tem a ver com a existência de uma certa inflação quando os técnicos ou operadores colocam o tempo gasto no pedido. Esse problema era notório na antiga aplicação de pedidos. Felizmente foi descontinuada e com a nova aplicação de pedidos GLPI diminuiu bastante.

Para terminar, os operadores e técnicos perdem imenso tempo a tratar pedidos via telefone. Devia apostar-se na *web* que demonstrou uma enorme eficácia em resolução de problemas / tempo gasto.

Trabalho futuro

O trabalho futuro passa pela consolidação do *data mart* dos pedidos e na implementação dos outros três *data marts*.

No *data mart* dos pedidos, existem muitos utilizadores dos quais não se conseguiu obter os dados relativos à faixa etária, género e nacionalidade. Seria importante consolidar essa informação, atualizando os sistemas operacionais ou tentar obter essa informação de outras fontes. Relativamente ao atributo “grupo” que neste momento só existe a opção “Técnicos” e “Operadores” obtidos na AD, poderia ser interessante estender a outros grupos de utilizadores (por exemplo grupos e investigação ou sub-áreas de departamentos ou unidades)

Nos *data marts* que impliquem tratar *logs*, será desafiante concentrá-los todos no DW e proceder ao seu tratamento de modo a prever todas as diferenças e exceções que existem. O volume de dados e o tempo de processamento do ETL também vão aumentar significativamente. Consequentemente será necessário efetuar alterações na arquitetura.

No *data mart* das redes será necessário compreender as origem de dados que são dados produzidos por ferramentas de monitorização e rede. Sendo muitas delas proprietárias, poderá ser complicado extrair informação.

Conseguir que as *pivot tables* do Excel acedam aos dados do cubo, seria uma mais valia, bem como estar atento a novas ferramentas analíticas que apareçam no mercado que sejam gratuitas e que consigam ligar-se ao Mondrian.

Bibliografia

[1] Ralph Kimball e Margy Ross, The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling, Wiley, 3ª edição, 2002, ISBN 978-1-118-53080-

[2] Ralph Kimball e Joe Caserta, The Data Warehouse ETL Toolkit, Wiley, 1ª edição, 2004, ISBN 074-645-67578

[3] Ralph Kimball et al, The Data Warehouse Lifecyle Toolkit, Wiley, 2ª edição, 2008, ISBN 978-047-014-977-5

[4] Data warehouse , conceitos e modelos com exemplos práticos, Carlos Pampulim caldeira edições sílabo, 1ª edição lisboa 2008, ISBN 978-972-618-479-9

[5] Sistemas de suporte à decisão, Bruno cortes, FCA, 1ª Edição ,junho 2005 ISBN 972-722-517-9

[6] Mondrian in Action, William D. Back et al, Hanning, 1ª Edição, 2014, ISBN 978-161-729-098-5

[7] Dimentional Data Warehousing with MySql, 1ª edição, 2007, ISBN 978-097-521-282-0

[8] Fast Track to MDX, Mark Whitehorn et al, Springer, 2ª Edição , 2006 ISBN 978-1-84628-174-7

[9] Data Mining pratical machine learning tools and techniques, Ian H.witten et al, Elsevier, 3ª Edição, 2011, ISBN 978-0-12-374856-0

[10] Documentação Online do Mondrian:
http://mondrian.pentaho.com/documentation/

[11] Documentação do Kettle:
http://wiki.pentaho.com/display/EAI/Spoon+User+Guide

[12] Weka:
http://www.cs.waikato.ac.nz/ml/weka/

[13] Tutorial MDX:
http://www.iccube.com/support/documentation/mdx_tutorial/gentle_introduction.php

[14] Blog sobre Mondiran as XMLA provider:
http://business-intelligence.phi-integration.com/2008/04/testing-mondrian-as-xmla-provider.html

[15] Video demonstrativo da ligação pivot tables ao Mondrian via XMLA:
https://www.youtube.com/watch?v=8eq_dE7_O3s

[16] JPivot:
http://jpivot.sourceforge.net/

[17] Saiku:
http://www.meteorite.bi/products/saiku

Anexo A - Analise da origem dos dados do OLTP

Gestão de Pedidos

O modelo de dados do qual se vai obter informação é:

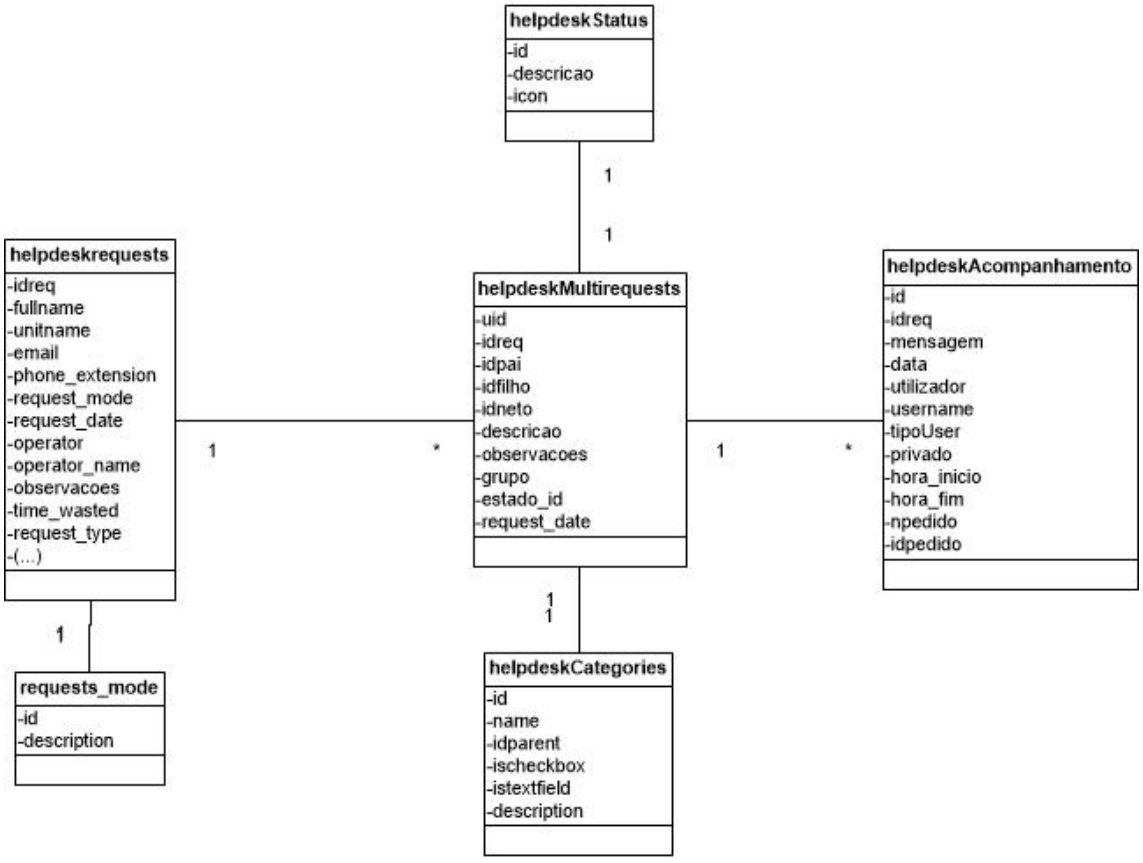


Figura 32 - Modelo de dados da Gestão de pedidos

Efetuando algumas *queries* de SQL foram encontradas as seguintes volumetrias:

Total de pedidos	6557
Sub-pedidos	6845
pedidos com mais que um sub-pedido	151
Pedidos de alunos	2634
Pedidos de Funcionários	3923

Tabela 64 - Contabilização dos pedidos

Embora a cada pedido possa estar associado vários sub-pedidos, o mais comum é que cada pedido só tenha um sub-pedido (só 151 dos 6557)

Também se pode constatar que embora haja mais alunos do que docentes e funcionários, estes fazem mais pedidos (3923 vs 2634)

Utilizadores Distintos	3218
Alunos	1872
Funcionários	1346
Max de pedidos de um utilizador	519
Min de pedidos por um utilizador	1
Media de pedidos por utilizador	2

Tabela 65 - Utilizadores

(*) O utilizador “suporte” tem 519 pedidos devido ao facto de muitas vezes se ser usado para registar operações internamente.

Existem 3218 utilizadores distintos que efetuaram pedidos, sendo 1872 alunos e o restante 1346. Normalmente um utilizador comum faz um ou dois pedidos.

Total de acompanhamentos	2384
Máximo d acompanhamentos num pedido	27
Min de acompanhamentos num pedido	0
Media de acompanhamentos	4.8

Tabela 66 - Contabilização de acompanhamentos

Acompanhamentos são ações que são registadas durante o tratamento de pedidos. Em média são precisas quase cinco interações para tratar um pedido

Natureza de pedidos	**Qtd**
Área do Utilizador	2648
Questões	1980
Outros Problemas	558

Instalação de Software	371
Serviços Web	326
Wireless	313
Suporte a Laboratório de Aulas	191
Instalação de Sistema Operativo	164
Configuração da VPN	76
Entrega de Material	72
Configuração Email	63
Problemas de Rede	36
Problemas de Hardware	29
Instalação de Hardware	18

Tabela 67 - Natureza de pedidos

Se os pedidos forem ordenados por naturezas, verifica-se que há muitos pedidos que caem em grupos mais generalistas. Logo a seguir vêm questões relacionadas com software (instalação e problemas com as aplicações do portal). A seguir, veem questões de infraestrutura e finalmente questões relacionadas com hardware.

Por estado	**Qtd**
Concluído	6838
Pendente	3
Pendente para entrega	4

Tabela 68 - Distribuição por estado

Como por norma os pedidos têm que ser resolvidos em 48 horas, constata-se que a maioria estão concluídos, restando apenas os que estão pendentes por terceiros.

Canal	
Presencial	3200
Web	2049
Email	1078
Telefone	518

Tabela 69 - Distribuição por canal

Podemos observar que os utilizadores preferem vir tratar das suas questões de forma presencial. Em segundo lugar fica a via eletrónica (web ou email). A menos utilizada é a via telefónica.

Análise dos dados:

Na tabela “helpdeskrequests” existe um campo para o nome do utilizador e outro para o departamento. No entanto, alguns de estes estão corrompidos por questões de *encoding*. Também se detetou poucos casos de utilizadores que foram associados erradamente a um departamento (por engano, ou porque entretanto mudou). Existem campos nulos ou

vazios que têm significado de negócio, nomeadamente na atribuição dos pedidos a um grupo ou operador. Neste caso o nulo, simplesmente significa que não foi atribuído a ninguém. Relativamente ás naturezas dos sub-pedidos, estas podem ter três níveis de profundidade, sendo o terceiro nível demasiado especifico e sem grande utilidade para o negócio.

Grupos

O modelo de dados do qual se vai obter informação é:

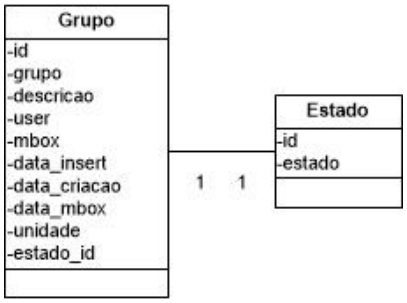


Figura 33 - Modelo de dados para a gestão de grupos

Algumas volumetrias:

Total	73
Utilizadores distintos	12
Max de pedidos por utilizadores	33
Min de pedidos por utilizadores	1
Média de pedidos por utilizadores	6

Tabela 70 - Volumetrias da gestão de grupos

Esta funcionalidade é pouco usada tenho apenas 73 pedidos e usada por doze utilizadores distintos.

Por estado	
Validado pelo Responsável	1
Aceite	64
Rejeitado	8

Tabela 71 - Distribuição por estado

A maioria dos pedidos foram aceites, tirando oito que foram rejeitados.

Análise dos dados

Esta é uma aplicação simples e com pouco uso e por esse motivo não foram encontradas anomalias

Tomadas

O modelo de dados do qual se vai obter informação é:

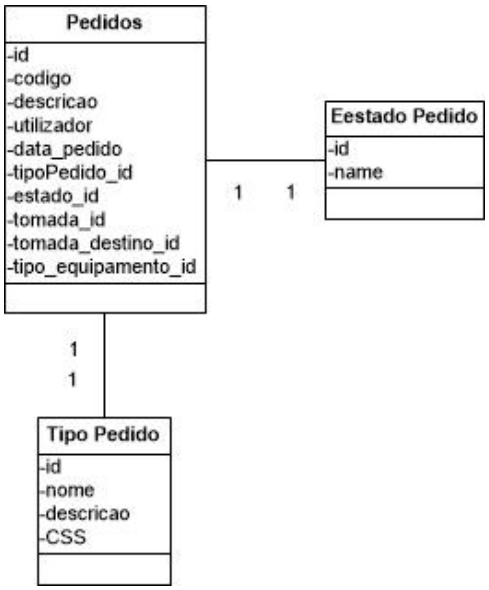


Figura 34 - Modelo de dados da aplicação das tomadas

Algumas volumetrias:

Total	868
Utilizadores distintos	257
Máximo pedidos por utilizador	47
Mínimo pedidos utilizador	1
Média de pedidos utilizador	3

Tabela 72 - Volumetrias da aplicação do site das tomadas

Por tipo de pedido	
Alteração de Equipamento	343
Ativação de Tomada	285
Mover Equipamento	125
Remoção de Tomada	115

Tabela 73 - Distribuição por operação

Por estado	
Novo	1
Para o suporte	1
Anulado	39
Tratado	827

Tabela 74 - Distribuição por estado

Análise aos dados

Não foram encontradas anomalias nos dados.

Inbox

O modelo de dados do qual se vai obter informação é:

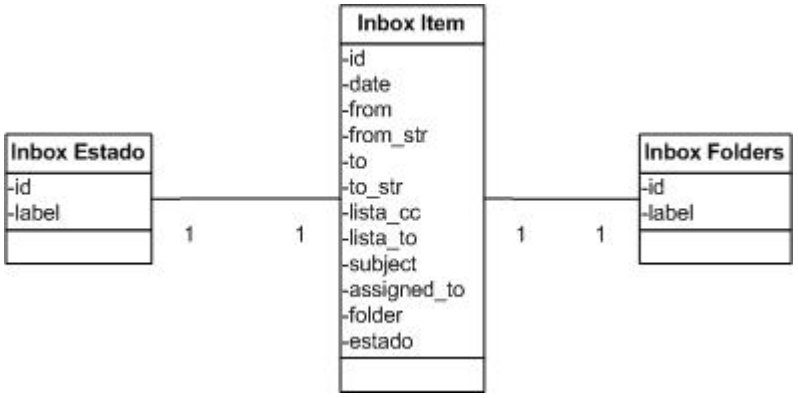


Figura 35 - Modelo de dados da Inbox

Volumetrias:

Total	79781
Utilizadores distintos	2381
Número máximo de pedidos por utilizador	69268
Número Mínimo de pedidos por utilizador	1

Este volume parece extremamente exagerado. Fazendo alguma investigação, constatou-se que havia muitos mails de MAILER-DEAMON ou internos do *exchange*:

- 'MicrosoftExchange??????????????????????@ul.pt',
- MAILER-DAEMON@??????01.fc.ul.pt
- MAILER-DAEMON@??????02.fc.ul.pt
- MAILER-DAEMON@??????03.fc.ul.pt

Nota: o nome dos servidores foram rasurados

Retirando esses emails :

Total	8768
Número máximo de pedidos por utilizador	603
Número Mínimo de pedidos por utilizador	1
Média de pedidos por utilizador	3,7

Figura 36 - Volumetrias da Inbux retirando mails incorrectos

Atribuídos a	
	3839
CI_OPERADORES_SUPORTE	1409
CI_TECNICOS_MOODLE	922
CI_TECNICOS_USID	505
CI_TECNICOS_REDE	428
CI_TECNICOS_SISTEMAS	400
xxxxxxxxxxxxxxxx	375
CI_TECNICOS_VC	272
xxxxxxxxxxxxxxxx	120
CI_TECNICOS_SUPORTE	47
xxxxxxxxxxxxxxxx	45
xxxxxxxxxxxxxxxx	45
xxxxxxxxxxxxxxxx	42
xxxxxxxxxxxxxxxx	40
xxxxxxxxxxxxxxxx	36
CI_COORDENACAO	33
xxxxxxxxxxxxxxxx	30
xxxxxxxxxxxxxxxx	29
xxxxxxxxxxxxxxxx	22
xxxxxxxxxxxxxxxx	21
xxxxxxxxxxxxxxxx	21
xxxxxxxxxxxxxxxx	18
CI_TECNICOS_REDES	15
xxxxxxxxxxxxxxxx	13
xxxxxxxxxxxxxxxx	12
xxxxxxxxxxxxxxxx	10
xxxxxxxxxxxxxxxx	10
xxxxxxxxxxxxxxxx	5
CI_OPERADORES_MOODLE	2
TO_DELETE	2

Figura 37 - Lista de emails mais comuns

Xxxxxxxxxxxxxxxx são pedidos que foram diretamente atribuídos a um operador ou técnico da UI. Por questões de privacidade esses dados foram rasurados.

Pasta de mail	
respondidos	3081
trash	2935
arquivo	1588
pedidos	897
enviados	239
delivery error	19
inbox	9

Pedidos GLPI

O modelo de dados do qual se vai obter informação é:

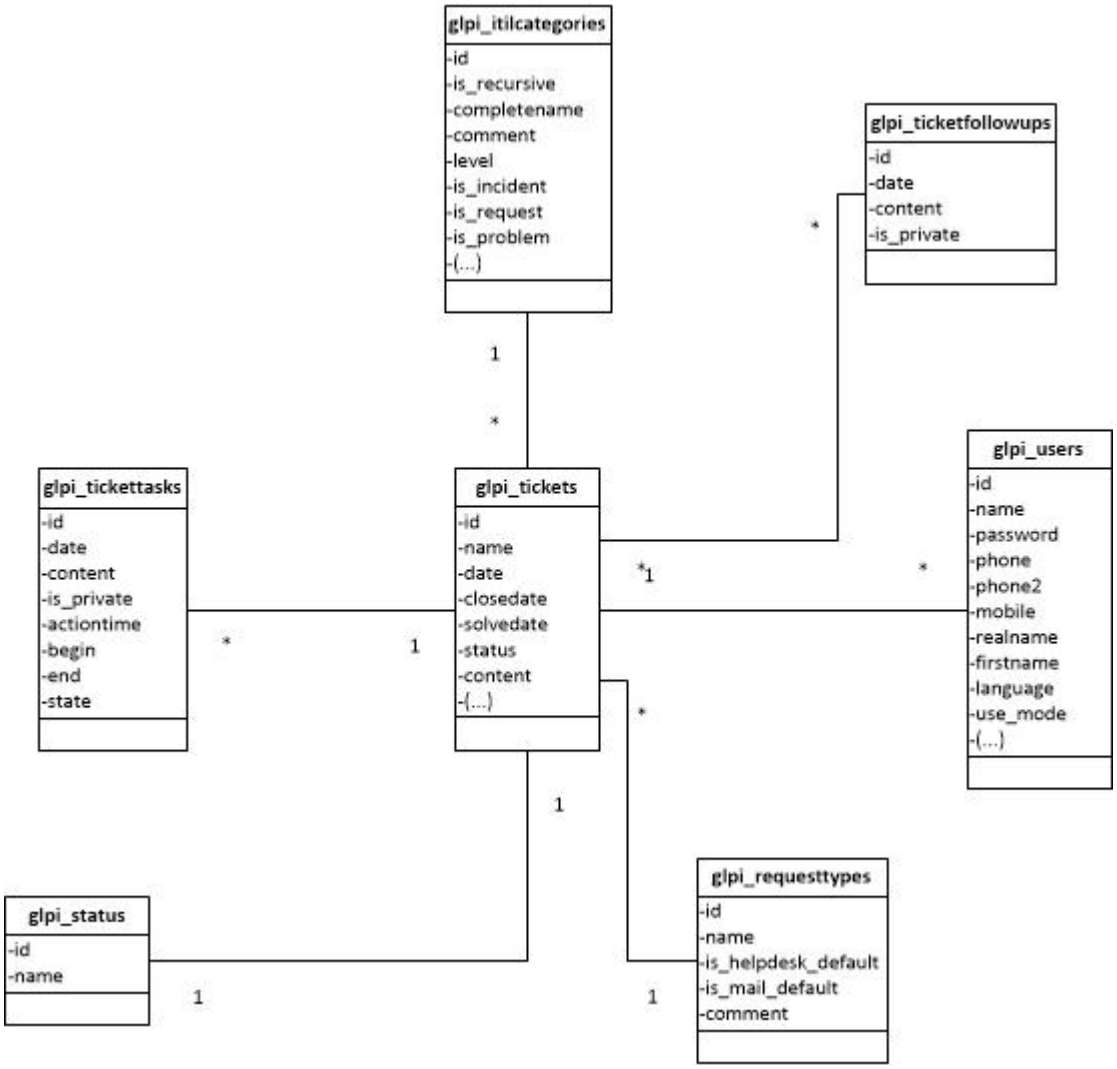


Figura 38 - Modelo de dados do GLPI

Total	751
Utilizadores distintos	545
Max de pedidos por utilizadores	48
Min de pedidos por utilizadores	1
media de pedidos por utilizadores	1.4

Aplicações

A origem dos dados são *logs* dos servidores web. Existe balanceamento de carga e existem vários servidores de produção. No entanto os *logs* estão centralizados num único servidor: o SYSLOG. É gerado um ficheiro de *logs* por dia. Por exemplo: 01-01-2015.log. Dentro desses ficheiros existem várias entradas com o seguinte formato:

Parte	**Descrição**
Data	2015-01-01
Hora	00:00:01
Aplicação	GestaoPedidos
User	rjsimoes
Role	0
IP	2001:690:21c0:f050::20
Acção	Inbox/mailsBeingRead
Metodos	POST|AJAX
UserAgent	Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36

Nas funcionalidades que são acedidas sem o utilizador fazer login o campo User não é preenchido.

O Role, o IP e Métodos não acrescentam nada ao DW.

Em relação à ação existem muitas que não têm qualquer utilidade e que devem ser descartadas. As que têm utilidade têm que ser mapeadas.

Os *logs* aplicacionais registam as invocações que foram feitas a esses servidores. Essas invocações podem resultar em operações de consulta, por exemplo: listagens, geração de documentos, etc ou operações de escrita , exemplo: inserir novo registo, apagar ou

editar. Nas operações de consulta, os dados dos *logs* são suficientes, já que é expresso a data, utilizador e a operação que foi mapeando o URL da invocação. No entanto, nas operações de escrita, não é bem assim, porque uma invocação de um URL de uma ação que se conhece ser de escrita, pode não resultar numa alteração da base de dados. Por exemplo, existe um formulário em que se preenche uns campos e carrega-se no botão "submeter". Esse botão despoleta uma ação e invoca uma chamada ao servidor "/inserePedido.php". No entanto existem validações *server-side* aos campos do formulário, o utilizador não preencheu devidamente um dos campos, o registo não é inserido e é devolvido uma lista de erros. Nos *logs* irá constar uma entrada em que na data D o utilizador U fez a invocação de "/inserePedido.php", mas essa invocação não se traduziu em alterações na BD.

Por outro lado, a informação do *user* só existe nos *logs*. Por esta razão será necessário cruzar a informação nas tabelas de registo das BD's relacionais com a informação dos l*ogs*.

Ao analisar os *logs* com algumas ferramentas como o "awk" e "grep" e criando pequenas aplicações JAVA para fazer *parsing* e inserir as entradas em BD, constatou-se que existe uma que não têm qualquer utilidade de negócio:

- Invocações de *keep-alive*:
- *Features* do portal da FCUL que não têm qualquer utilidade de negócio
- Invocações Ajax para preencher listas de valores para componentes de formulários
- Erros e *warnings*

Será importante eliminar à partida estas entradas para não entrarem no fluxo de processos do ETL e estar a tratar, a validar e a processar linhas que sabe-a priori que não têm qualquer utilidade.

Por outro lado, será preciso mapear as operações registadas na BD relacional com as operações de escrita.

No futuro será necessário selecionar as aplicações que vão ser cruzadas com a informação dos l*ogs.*

Sistemas

A análise aos logs dos servidores web que foi efetuado para Aplicações também é válida para sistemas. Outros logs de outros servidores foram mostrados por alto e

demonstraram que existem muitos formatos e que poderá ser complexo fazer *parsing* a todos eles.

Anexo B – Listagem de ferramentas para BI

Produto	URL	Observações
BEE / gooddata	http://www.gooddata.com/	Plataforma BI em cloud
BIRT	http://www.eclipse.org/birt/	Ferramente de reporting para BI que é um plugin do eclipes
jasperSoft	http://www.jaspersoft.com/business-intelligence-solutions	Reporting e Business analyticas nanlytics
opernI	http://openi.org/	Analitics orientado para o bigdata
Pentaho	http://www.pentaho.com/	Soluções integradas BI
Mondrian	http://community.pentaho.com/projects/119ondrian/	Versão Lite e gratuita do Pentaho
Haddop	https://hadoop.apache.org/	Procressamento distribuido
jpivot	http://jpivot.sourceforge.net/	Frontend em JSP que gera uma pivotTable e que usa o mondrian
Kettle	http://community.pentaho.com/projects/data-integration/	Ferramenta de Data Integration da Pentaho.
Claudera	http://www.cloudera.com	Ceder aos dados do hadoop
Hive	https://hive.apache.org/	
hortonworks	http://hortonworks.com/	Soluções integradas BI baseadas em Haddop
Saiku	http://wiki.meteorite.bi/display/SAIK/Saiku	Ferramenta analítica baseada na web.
Kylin	http://kylin.incubator.apache.org/index.html	OLAP Engine Big Data
Olap4j	http://www.olap4j.org/	Cubo Olap para JAVA
PhpMyOlap	http://sourceforge.net/projects/phpmyolap/?source=directory	Cubo olap para PHP
JaspersoftETL	https://community.jaspersoft.com/project/jaspersoft-etl	Ferramenta de ETL

Anexo C - Ficheiro ARFF para o WEKA

```
@relation pedidos

@attribute CANAL_desc {Presencial, web, E-mail, Telefone}

@attribute DATA_ano NUMERIC

@attribute DATA_mes { Abril,Agosto,Dezembro,Fevereiro,Janeiro,Julho,Junho,Maio,Março,Novembro,Outubro,Setembro}

@attribute DATA_semestre {S1,S2,F }

@attribute DATA_tipo_dia {Dia_semana,Feriado,Fim_de_semana }

@attribute RELOGIO_periodo_dia {Manhã,Noite,Tarde}

@attribute OPERACAO_aplicacao {Gestao_Grupos, Inbox,Pedidos,Pedidos_GLPI,Tomadas}

@attribute OPERACAO_categoria

{CONTAS,DESENVOLVIMENTO,HARDWARE,OUTRAS,REDES,SISTEMAS,SOFTWARE,SUPORTE_LABS}

@attribute OPERACAO_TIPO string

@attribute ESTADO_desc { CONCLUIDO,EM_CURSO,NOVO,OUTRO,PENDENTE}

@attribute USER_Departamento string

@attribute USER_faixa_etaria {Desconhecido,[18-20],[21-23],[24-26],[27-30],[31-40],[41-50],[51-65],[65+]}

@attribute USER_grupo { Nenhum,Operador,Técnico}

@attribute USER_genero {Desconhecido,Feminino,Masculino}

@attribute USER_tipo_utilizador {ALUNO,DOCENTE,EXTERNO,FUNCIONARIO}

@attribute USER_nacionalidade string

@attribute MEDIDA_acompanhamentos NUMERIC

@attribute MEDIDA_tempoGasto NUMERIC

 @DATA

Presencial,2015,Janeiro,S1,Dia_semana,Manhã,Pedidos,CONTAS,'Área do Utilizador',CONCLUIDO,'Departamento de

Informática',[21-23],Nenhum,Masculino,ALUNO,'Portugal','0','1'

(…)
```

Anexo D - Resultados dos algoritmos de Classificação do weka

Algoritmo J48

```
=== Summary ===

Correctly Classified Instances        1864               82.8444 %
Incorrectly Classified Instances       386               17.1556 %
Kappa statistic                          0.7128
Mean absolute error                      0.0734
Root mean squared error                  0.1966
Relative absolute error                 35.5989 %
Root relative squared error             61.1064 %
Total Number of Instances             2250


=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         ?
               0,936    0,207    0,844      0,936    0,887      0,742    0,959     0,964     [0-10]
               0,726    0,064    0,802      0,726    0,762      0,685    0,947     0,858     [10-30]
               0,572    0,002    0,950      0,572    0,714      0,723    0,917     0,708     [30-60]
               0,441    0,000    1,000      0,441    0,612      0,656    0,883     0,555     [60-120]
               0,874    0,030    0,707      0,874    0,781      0,766    0,960     0,803     [120+]
Weighted Avg.  0,828    0,132    0,837      0,828    0,822      0,724    0,950     0,888
```

```
=== Confusion Matrix ===

   a   b   c   d   e   f   <-- classified as
   0   0   0   0   0   0 |   a = ?
   0 1146  63   1   0  15 |   b = [0-10]
   0 151 430   1   0  10 |   c = [10-30]
   0  32  22  95   0  17 |   d = [30-60]
   0  22   8   1  41  21 |   e = [60-120]
   0   7  13   2   0 152 |   f = [120+]
```

Algoritmo Random Tree

```
               === Summary ===

Correctly Classified Instances        1753               77.9111 %
Incorrectly Classified Instances       497               22.0889 %
Kappa statistic                          0.6384
Mean absolute error                      0.0775
Root mean squared error                  0.2562
Relative absolute error                 37.5604 %
Root relative squared error             79.6153 %
Total Number of Instances             2250

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         ?
               0,873    0,201    0,838      0,873    0,855      0,675    0,874     0,854     [0-10]
               0,688    0,109    0,692      0,688    0,690      0,580    0,830     0,626     [10-30]
               0,633    0,017    0,750      0,633    0,686      0,666    0,839     0,572     [30-60]
               0,484    0,014    0,600      0,484    0,536      0,521    0,746     0,337     [60-120]
               0,730    0,022    0,738      0,730    0,734      0,712    0,877     0,607     [120+]
Weighted Avg.  0,779    0,142    0,776      0,779    0,777      0,646    0,855     0,733
```

```
=== Confusion Matrix ===

   a   b   c   d   e   f   <-- classified as
   0   0   0   0   0   0 |   a = ?
   0 1069 124  14   6  12 |   b = [0-10]
   0 151 407  15   8  11 |   c = [10-30]
   0  25  19 105   4  13 |   d = [30-60]
   0  16  19   4  45   9 |   e = [60-120]
   0  14  19   2  12 127 |   f = [120+]
```

Algoritmo Random Forest

```
=== Summary ===

Correctly Classified Instances        1843               81.9111 %
Incorrectly Classified Instances       407               18.0889 %
Kappa statistic                          0.7035
Mean absolute error                      0.0793
Root mean squared error                  0.1998
Relative absolute error                 38.4693 %
Root relative squared error             62.1027 %
Total Number of Instances             2250


=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,000    0,000    0,000     0,000    0,000     0,000    ?        ?        ?
               0,901    0,153    0,875     0,901    0,888     0,751    0,962    0,972    [0-10]
               0,779    0,101    0,734     0,779    0,756     0,666    0,944    0,870    [10-30]
               0,596    0,013    0,786     0,596    0,678     0,663    0,943    0,746    [30-60]
               0,473    0,006    0,759     0,473    0,583     0,586    0,901    0,597    [60-120]
```

```
               0,776    0,020    0,763     0,776    0,769     0,750    0,969     0,857    [120+]
Weighted Avg.  0,819    0,113    0,818     0,819    0,816     0,715    0,954     0,904


=== Confusion Matrix ===


   a   b   c   d   e   f   <-- classified as
   0   0   0   0   0   0 |   a = ?
   0 1104  96  10   3  12 |   b = [0-10]
   0  111 461  12   2   6 |   c = [10-30]
   0   18  33  99   2  14 |   d = [30-60]
   0   18  17   4  44  10 |   e = [60-120]
   0   10  21   1   7 135 |   f = [120+]
```

Algoritmo JRip

```
=== Summary ===


Correctly Classified Instances        1823               81.0222 %
Incorrectly Classified Instances       427               18.9778 %
Kappa statistic                          0.669
Mean absolute error                      0.102
Root mean squared error                  0.2285
Relative absolute error                 49.4558 %
Root relative squared error             71.0182 %
Total Number of Instances             2250


=== Detailed Accuracy By Class ===


               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
```

```
          0,000   0,000   0,000    0,000   0,000    0,000   ?      ?      ?
          0,960   0,324   0,780    0,960   0,861    0,674   0,827    0,777    [0-10]
          0,644   0,041   0,849    0,644   0,732    0,664   0,838    0,721    [10-30]
          0,542   0,000   1,000    0,542   0,703    0,723   0,843    0,639    [30-60]
          0,430   0,000   1,000    0,430   0,602    0,648   0,792    0,498    [60-120]
          0,782   0,013   0,834    0,782   0,807    0,792   0,920    0,766    [120+]
Weighted Avg.   0,810   0,188   0,827    0,810   0,800    0,683   0,837    0,740


=== Confusion Matrix ===


   a   b   c   d   e   f   <-- classified as
   0   0   0   0   0   0 |   a = ?
   0 1176  45   0   0   4 |   b = [0-10]
   0  208 381   0   0   3 |   c = [10-30]
   0   55  12  90   0   9 |   d = [30-60]
   0   38   4   0  40  11 |   e = [60-120]
   0   31   7   0   0 136 |   f = [120+]
```

Algoritmo PART

```
=== Summary ===


Correctly Classified Instances        1832               81.4222 %
Incorrectly Classified Instances       418               18.5778 %
Kappa statistic                          0.6977
Mean absolute error                      0.0735
Root mean squared error                  0.2136
Relative absolute error                 35.6414 %
Root relative squared error             66.3873 %
```

```
Total Number of Instances           2250


=== Detailed Accuracy By Class ===


               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         ?
               0,888    0,144    0,880      0,888    0,884      0,744    0,938     0,948     [0-10]
               0,785    0,102    0,733      0,785    0,759      0,669    0,919     0,798     [10-30]
               0,590    0,007    0,875      0,590    0,705      0,702    0,869     0,698     [30-60]
               0,430    0,002    0,889      0,430    0,580      0,608    0,808     0,536     [60-120]
               0,810    0,039    0,632      0,810    0,710      0,689    0,906     0,715     [120+]
Weighted Avg.  0,814    0,109    0,822      0,814    0,812      0,712    0,920     0,855


=== Confusion Matrix ===


   a   b   c   d   e   f   <-- classified as
   0   0   0   0   0   0 |   a = ?
   0 1088 103   5   1  28 |   b = [0-10]
   0 106 465   6   1  14 |   c = [10-30]
   0  17  29  98   2  20 |   d = [30-60]
   0  12  18   3  40  20 |   e = [60-120]
   0  13  19   0   1 141 |   f = [120+]
```

Algoritmo IBK

```
=== Summary ===


Correctly Classified Instances        1772               78.7556 %
Incorrectly Classified Instances       478               21.2444 %
```

```
Kappa statistic                          0.65
Mean absolute error                      0.0773
Root mean squared error                  0.2496
Relative absolute error                 37.4624 %
Root relative squared error             77.5749 %
Total Number of Instances             2250

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         ?
               0,891    0,187    0,850      0,891    0,870      0,708    0,921     0,923     [0-10]
               0,742    0,106    0,715      0,742    0,728      0,629    0,893     0,731     [10-30]
               0,530    0,019    0,688      0,530    0,599      0,577    0,835     0,479     [30-60]
               0,366    0,017    0,479      0,366    0,415      0,397    0,783     0,316     [60-120]
               0,690    0,016    0,779      0,690    0,732      0,712    0,898     0,662     [120+]
Weighted Avg.  0,788    0,133    0,782      0,788    0,783      0,665    0,900     0,795

=== Confusion Matrix ===

   a   b   c   d   e   f   <-- classified as
   0   0   0   0   0   0 |   a = ?
   0 1091 102  14   8  10 |   b = [0-10]
   0 128 439  12   5   8 |   c = [10-30]
   0  21  35  88  14   8 |   d = [30-60]
   0  18  20  13  34   8 |   e = [60-120]
   0  25  18   1  10 120 |   f = [120+]
```

Algoritmo KStar

```
=== Summary ===

Correctly Classified Instances        1797               79.8667 %
Incorrectly Classified Instances       453               20.1333 %
Kappa statistic                          0.6694
Mean absolute error                      0.0798
Root mean squared error                  0.2178
Relative absolute error                 38.6978 %
Root relative squared error             67.6757 %
Total Number of Instances             2250

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         ?
               0,891    0,165    0,866      0,891    0,878      0,728    0,953     0,965     [0-10]
               0,769    0,116    0,703      0,769    0,734      0,635    0,923     0,794     [10-30]
               0,536    0,017    0,718      0,536    0,614      0,595    0,923     0,657     [30-60]
               0,419    0,013    0,574      0,419    0,484      0,472    0,896     0,517     [60-120]
               0,707    0,013    0,815      0,707    0,757      0,740    0,973     0,844     [120+]
Weighted Avg.  0,799    0,123    0,796      0,799    0,795      0,684    0,942     0,869

=== Confusion Matrix ===

   a   b   c   d   e   f   <-- classified as
   0   0   0   0   0   0 |   a = ?
   0 1091 111  14   4   5 |   b = [0-10]
   0 115 455  15   3   4 |   c = [10-30]
```

```
  0   20   37   89   11    9 |    d = [30-60]

  0   16   22    6   39   10 |    e = [60-120]

  0   18   22    0   11  123 |    f = [120+]
```

Algoritmo BayesNet

```
=== Summary ===



Correctly Classified Instances        1628               72.3556 %

Incorrectly Classified Instances      622                27.6444 %

Kappa statistic                       0.5774

Mean absolute error                   0.1069

Root mean squared error               0.2608

Relative absolute error               51.8266 %

Root relative squared error           81.032  %

Total Number of Instances             2250



=== Detailed Accuracy By Class ===



               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class

               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         ?

               0,709    0,087    0,907      0,709    0,796      0,627    0,907     0,930     [0-10]

               0,834    0,230    0,565      0,834    0,673      0,546    0,870     0,611     [10-30]

               0,584    0,015    0,752      0,584    0,658      0,640    0,912     0,687     [30-60]

               0,570    0,044    0,361      0,570    0,442      0,424    0,892     0,499     [60-120]

               0,661    0,013    0,816      0,661    0,730      0,715    0,968     0,835     [120+]

Weighted Avg.  0,724    0,112    0,776      0,724    0,734      0,605    0,902     0,803



=== Confusion Matrix ===
```

```
  a   b   c   d   e   f  <-- classified as
  0   0   0   0   0   0 |  a = ?
  0 869 294  19  42   1 |  b = [0-10]
  0  65 494  10  15   8 |  c = [10-30]
  0  14  37  97  10   8 |  d = [30-60]
  0   7  21   3  53   9 |  e = [60-120]
  0   3  29   0  27 115 |  f = [120+]
```

Algoritmo NaiveBayes

```
=== Summary ===

Correctly Classified Instances        1673               74.3556 %
Incorrectly Classified Instances       577               25.6444 %
Kappa statistic                          0.6012
Mean absolute error                      0.1022
Root mean squared error                  0.2504
Relative absolute error                 49.5424 %
Root relative squared error             77.8258 %
Total Number of Instances             2250


=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,000    0,000    0,000      0,000    0,000      0,000    ?         ?         ?
               0,754    0,107    0,894      0,754    0,818      0,647    0,907     0,931     [0-10]
               0,806    0,201    0,589      0,806    0,680      0,555    0,875     0,633     [10-30]
               0,566    0,019    0,701      0,566    0,627      0,604    0,891     0,680     [30-60]
               0,527    0,029    0,441      0,527    0,480      0,458    0,893     0,469     [60-120]
```

```
          0,741    0,015    0,801     0,741    0,770     0,752    0,974     0,866     [120+]
Weighted Avg.    0,744    0,115    0,773     0,744    0,750     0,620    0,902     0,810


=== Confusion Matrix ===


  a   b   c   d   e   f   <-- classified as
  0   0   0   0   0   0 |   a = ?
  0 924 250  23  25   3 |   b = [0-10]
  0  86 477  11   9   9 |   c = [10-30]
  0  13  43  94   7   9 |   d = [30-60]
  0   7  20   6  49  11 |   e = [60-120]
  0   4  20   0  21 129 |   f = [120+]
```

Algoritmo Voting

```
=== Summary ===


Correctly Classified Instances        1859               82.6222 %
Incorrectly Classified Instances       391               17.3778 %
Kappa statistic                          0.7157
Mean absolute error                      0.0579
Root mean squared error                  0.2407
Relative absolute error                 28.086  %
Root relative squared error             74.7928 %
Total Number of Instances             2250


=== Detailed Accuracy By Class ===


               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
```

```
               0,000   0,000   0,000    0,000   0,000    0,000   ?       ?       ?
               0,905   0,144   0,882    0,905   0,894    0,763   0,880    0,850    [0-10]
               0,789   0,094   0,750    0,789   0,769    0,684   0,847    0,647    [10-30]
               0,590   0,008   0,852    0,590   0,698    0,691   0,791    0,533    [30-60]
               0,441   0,005   0,788    0,441   0,566    0,577   0,718    0,371    [60-120]
               0,828   0,028   0,709    0,828   0,764    0,745   0,900    0,600    [120+]
Weighted Avg.  0,826   0,106   0,828    0,826   0,823    0,728   0,860    0,734


=== Confusion Matrix ===


   a    b    c    d    e    f   <-- classified as
   0    0    0    0    0    0 |    a = ?
   0 1109   94    5    2   15 |    b = [0-10]
   0  109  467    7    1    8 |    c = [10-30]
   0   17   30   98    3   18 |    d = [30-60]
   0   14   16    4   41   18 |    e = [60-120]
   0    8   16    1    5  144 |    f = [120+]
```

Anexo E – Schema do Mondrian

```
<Schema name="Pedidos">
	<Dimension type="StandardDimension" name="Canal">
		<Hierarchy name="Canal" hasAll="true" allMemberName="Todos"  primaryKey="ID">
			<Table name="dim_canal">
			</Table>
			<Level name="Canal" table="dim_canal" column="Canal" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
			</Level>
		</Hierarchy>
	</Dimension>
	<Dimension type="TimeDimension" name="Data">
		<Hierarchy name="Lectivo" hasAll="true" allMemberName="Todos" primaryKey="id">
			<Table name="dim_data">
			</Table>
			<Level name="Ano" table="dim_data" column="Ano" type="Numeric" uniqueMembers="true"
levelType="TimeYears" hideMemberIf="Never">
			</Level>
			<Level name="Semestre" table="dim_data" column="Semestre" type="String" uniqueMembers="false"
levelType="TimeQuarters" hideMemberIf="Never">
			</Level>
			<Level name="Mes" table="dim_data" column="Descri&#231;&#227;o do m&#234;s" type="String"
uniqueMembers="false" levelType="TimeMonths" hideMemberIf="Never">
			</Level>
			<Level name="Dia" table="dim_data" column="Dia" type="String" uniqueMembers="false"
levelType="TimeDays" hideMemberIf="Never">
			</Level>
		</Hierarchy>
		<Hierarchy name="Legal" hasAll="true">
			<Table name="dim_data">
			</Table>
			<Level name="Ano" table="dim_data" column="Ano" type="Numeric" uniqueMembers="false"
levelType="TimeYears" hideMemberIf="Never">
			</Level>
			<Level name="M&#234;s" table="dim_data" column="Descri&#231;&#227;o do m&#234;s"
ordinalColumn="M&#234;s" type="String" uniqueMembers="false" levelType="TimeMonths" hideMemberIf="Never">
			</Level>
			<Level name="Dia" table="dim_data" column="Dia" type="Numeric" uniqueMembers="false"
levelType="TimeDays" hideMemberIf="Never">
			</Level>
		</Hierarchy>
		<Hierarchy name="Tipo de dia" hasAll="true" allMemberName="Todos" primaryKey="id">
			<Table name="dim_data">
			</Table>
			<Level name="Tipo de dia" table="dim_data" column="Tipo de dia" type="String" uniqueMembers="false"
levelType="TimeDays" hideMemberIf="Never">
			</Level>
			<Level name="Dia da semana" table="dim_data" column="Dia da Semana"  type="String"
uniqueMembers="false" levelType="TimeDays" hideMemberIf="Never">
			</Level>
		</Hierarchy>
	</Dimension>
	<Dimension type="StandardDimension" name="Estado">
		<Hierarchy name="Estado" hasAll="true" allMemberName="Todos" primaryKey="ID">
			<Table name="dim_estado">
			</Table>
			<Level name="Estado generico" table="dim_estado" column="Categoria" type="String"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
			</Level>
			<Level name="Estado" table="dim_estado" column="Estado" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
```

```
                    </Level>
                </Hierarchy>
        </Dimension>
        <Dimension type="StandardDimension" name="Relogio">
                <Hierarchy hasAll="true" allMemberName="Todos" primaryKey="ID">
                    <Table name="dim_relogio">
                    </Table>
                    <Level approxRowCount="" name="Periodo" table="dim_relogio" column="Periodo do dia" type="String"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="Hora" table="dim_relogio" column="Hora" ordinalColumn="Hora" type="Numeric"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="Minuto" table="dim_relogio" column="Minuto" ordinalColumn="Minuto" type="Numeric"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                </Hierarchy>
        </Dimension>
        <Dimension type="StandardDimension" name="Operacao">
                <Hierarchy name="Por aplica&#231;&#227;o" hasAll="true" allMemberName="Todos" primaryKey="ID">
                    <Table name="dim_operacao">
                    </Table>
                    <Level name="Aplica&#231;&#227;o" table="dim_operacao" column="Aplica&#231;&#227;o" type="String"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="Tipo de Opera&#231;&#227;o" table="dim_operacao" column="Tipo de Opera&#231;&#227;o"
type="String" uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="Sub-Tipo de Opera&#231;&#227;o" table="dim_operacao" column="Sub-Tipo de
Opera&#231;&#227;o" type="String" uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                </Hierarchy>
                <Hierarchy name="por Tipo" hasAll="true" allMemberName="Todos" primaryKey="ID">
                    <Table name="dim_operacao">
                    </Table>
                    <Level name="Categoria" table="dim_operacao" column="Categoria" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="Tipo de Opera&#231;&#227;o" table="dim_operacao" column="Tipo de Opera&#231;&#227;o"
type="String" uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="Sub-Tipo de Opera&#231;&#227;o" table="dim_operacao" column="Sub-Tipo de
Opera&#231;&#227;o" type="String" uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                </Hierarchy>
        </Dimension>
        <Dimension type="StandardDimension" name="Utilizador">
                <Hierarchy name="Departamento" hasAll="true" allMemberName="Todos" primaryKey="ID">
                    <Table name="dim_utilizador">
                    </Table>
                    <Level name="Departamento" table="dim_utilizador" column="Departamento" type="String"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="Grupo" table="dim_utilizador" column="Grupo" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="Utilizador" table="dim_utilizador" column="Nome" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
                    </Level>
                </Hierarchy>
                <Hierarchy name="Dados Utilizador" hasAll="true" allMemberName="false" primaryKey="ID">
                    <Table name="dim_utilizador">
                    </Table>
                    <Level name="Nacionalidade" table="dim_utilizador" column="Nacionalidade" type="String"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="G&#233;nero" table="dim_utilizador" column="G&#233;nero" type="String"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="Faixa Et&#225;ria" table="dim_utilizador" column="Faixa Et&#225;ria" type="String"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                    </Level>
                    <Level name="Utilizador" table="dim_utilizador" column="Nome" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
                    </Level>
```

```
                </Hierarchy>
                <Hierarchy name="G&#233;nero" hasAll="true" allMemberName="Todos" primaryKey="ID">
                        <Table name="dim_utilizador">
                        </Table>
                        <Level name="G&#233;nero" table="dim_utilizador" column="G&#233;nero" type="String"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                        </Level>
                        <Level name="Utilizador" table="dim_utilizador" column="Nome" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
                        </Level>
                </Hierarchy>
                <Hierarchy name="Faixa Et&#225;ria" hasAll="true" allMemberName="Todos" primaryKey="ID">
                        <Table name="dim_utilizador">
                        </Table>
                        <Level name="Faixa Et&#225;ria" table="dim_utilizador" column="Faixa Et&#225;ria" type="String"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                        </Level>
                        <Level name="Utilizador" table="dim_utilizador" column="Nome" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
                        </Level>
                </Hierarchy>
                <Hierarchy name="Tipo de Utilizador" hasAll="true" allMemberName="Todos" primaryKey="ID">
                        <Table name="dim_utilizador">
                        </Table>
                        <Level name="Tipo" table="dim_utilizador" column="Tipo de Utilizador" type="String"
uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
                        </Level>
                        <Level name="Utilizador" table="dim_utilizador" column="Nome" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
                        </Level>
                </Hierarchy>
        </Dimension>
        <Dimension type="StandardDimension" name="Operador">
                <Hierarchy name="Grupo" hasAll="true" allMemberName="Todos">
                        <Table name="dim_operador">
                        </Table>
                        <Level name="Grupo" table="dim_operador" column="Grupo" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
                        </Level>
                        <Level name="Utilizador" table="dim_operador" column="Nome" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
                        </Level>
                </Hierarchy>
        </Dimension>
        <Cube name="Pedidos" cache="true" enabled="true">
                <Table name="fact">
                </Table>
                <DimensionUsage source="Canal" name="Canal" foreignKey="dim_canal_id">
                </DimensionUsage>
                <DimensionUsage source="Data" name="Data" foreignKey="dim_data_id">
                </DimensionUsage>
                <DimensionUsage source="Relogio" name="Relogio" foreignKey="dim_relogio_id">
                </DimensionUsage>
                <DimensionUsage source="Estado" name="Estado" foreignKey="dim_estado_id">
                </DimensionUsage>
                <DimensionUsage source="Operacao" name="Operacao" foreignKey="dim_operacao_id">
                </DimensionUsage>
                <DimensionUsage source="Utilizador" name="Utilizador" foreignKey="dim_utilizador_id">
                </DimensionUsage>
                <DimensionUsage source="Operador" name="Operador" foreignKey="dim_operador_id">
                </DimensionUsage>
                <Measure name="Tempo Gasto" column="Tempo Gasto" aggregator="sum" visible="true">
                </Measure>
                <Measure name="Acompanhamentos" column="Acompanhamentos" aggregator="sum" visible="true">
                </Measure>
                <Measure name="Contagem" column="ID" aggregator="count" visible="true">
                </Measure>
        </Cube>
        <Role name="Tudo">
        </Role>
</Schema>
```

Anexo F – Formatar tempo gasto no MDX

```
 WITH
MEMBER measures.tempo as
 STR(Int(Measures.[Tempo Gasto]/60)) || " h " ||
 STR(
int(Measures.[Tempo Gasto] - Int(Measures.[Tempo Gasto]/60)*60 )
 ) || " m "
SET [~ROWS] AS
    {[Utilizador.Tipo de Utilizador].[Tipo].Members}
SELECT
NON EMPTY { [Measures].[tempo]} ON COLUMNS,
NON EMPTY [~ROWS] ON ROWS
FROM [Pedidos]
```