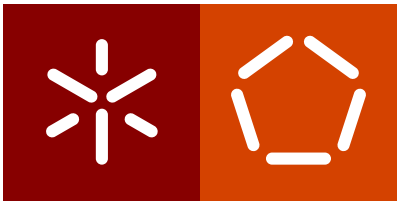
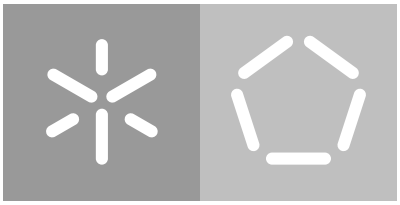


Universidade do Minho
Escola de Engenharia
Departamento de Informática

Nuno Miguel Monteiro Soares Dias

Deteção de Pontos Negros em Sistemas de ETL

October 2017



Universidade do Minho
Escola de Engenharia
Departamento de Informática

Nuno Miguel Monteiro Soares Dias

Deteção de Pontos Negros em Sistemas de ETL

Master dissertation
Master Degree in Computer Science

Dissertation supervised by
Orlando Belo

October 2017

Agradecimentos

Sendo a dissertação de mestrado um processo, na sua maioria, solitário, tal trabalho não é possível sem o contributo e experiência do orientador nem tão pouco sem o apoio dos amigos e familiares mais próximos.

Ao Professor Doutor Orlando Belo, orientador desta dissertação, agradeço o total apoio e disponibilidade que sempre o acompanharam, tendo o seu contributo sido essencial para que a dissertação se desenrolasse dentro dos objetivos previstos e, sobretudo, para que fosse bem-sucedida. Além disso, a sua elevada experiência foi muito importante, acalmando os ímpetos de um orientando, por vezes, ansioso e demasiado preocupado.

Aos meus amigos mais próximos, sobretudo ao Carlos agradeço pelo apoio prestado, mesmo que à distância de centenas de quilómetros. À minha amiga Cátia por ter estado sempre presente ao longo de todos estes meses de trabalho.

À Professora Teresa Azeredo e à filha, Marta, pelo contributo de vital importância nas temáticas ligadas à Matemática que indiretamente estão envolvidas na minha dissertação.

À minha família, sobretudo à minha Mãe pela paciência, sabedoria e sensatez que sempre a acompanha nas alturas de particular dificuldade.

Ao meu irmão pela sua atitude de constante desafio para que vá mais além no aprofundar do meu conhecimento e pelo seu espírito crítico apurado que tantas vezes me fez rever pormenores de pensamento.

À equipa "MyDO" que serviu, por diversas vezes, como escape aos problemas normais de uma dissertação.

A todos aqueles que direta ou indiretamente me acompanharam neste caminho o meu profundo e sincero agradecimento.

Resumo

Os sistemas de povoamento de *data warehouses*, vulgarmente designados por sistema de ETL – *Extract-Transform-Load* –, constituem a base de qualquer sistema de *data warehousing*. No entanto, poucas são as vezes em que a sua implementação ocorre de uma forma linear, metódica, seguindo um dado modelo de trabalho devidamente comprovado. Usualmente, estes sistemas estabelecem uma "ponte" entre os sistemas operacionais, muitas vezes de natureza diversa, e os sistemas de *data warehousing*, de forma a que seja possível assegurar o povoamento dos seus *data warehouses*, de uma forma regular e atual. Como tal, é muito normal terem que lidar com um volume de dados considerável e envolvendo processos de tratamento bastante complexos. Esses processos, que representam trabalho extra para o ETL, só são necessários devido à da elevada permeabilidade dos sistemas operacionais que facilitam a ocorrência de fenómenos de inconsistência e de omissão de valores. Para que tal não aconteça, as atuais técnicas e modelos de implementação baseados em processos típicos de "tentativa-erro" deverão ser abandonados desde início, dando lugar a uma arquitetura pensada com vista num melhor desempenho evitando, assim, situações em que um aumento no volume de dados do processo, tende a revelar um efeito "bola de neve" em termos do nível de *performance* do sistema. Neste trabalho de dissertação desenvolvemos uma técnica baseada em *process mining* que, recorrendo aos registos de execução detalhados de um processo ETL - *logs* -, permite descobrir todo o processo ETL a montante. Na posse dos dados relativos a cada passo de execução do processo ETL (tempo médio de execução, frequência absoluta, etc), podemos definir um modelo matemático que ilustra o "bem-estar", ou seja, o desempenho do nosso sistema através da correlação de todas estas variáveis. Desta forma, ao torná-lo acessível aos administradores dos sistemas, introduzimos um novo paradigma no desenvolvimento e manutenção de processos ETL, mais preocupado com questões como a *performance* ou um conhecimento mais aprofundado do impacto das decisões arquiteturais que são tomadas, nomeadamente a nível da escolha de componentes para executar cada passo do nosso ETL.

Abstract

ETL – Extract – Transform – Load – systems is the common name for the systems behind the data warehouses' populating process. In fact, they're the core piece of any data warehousing system. However, most of the times its implementation does not occur in a regular way. Usually, these systems establish the "bridge" between the operational environment, most of the times a heterogeneous one, so that its populating process proceeds in a regular and up to date way. Therefore, it's normal for these processes to cope with large volumes of data involving complex validation processes. These validation processes, that represent an extra effort for the ETL, are only necessary thanks to a high permeability of the operational systems, that facilitates the occurrence of value omissions or inconsistencies. In order to reverse the situation, the current ad-hoc technique must be abandoned from the very beginning, leaving place to a new one, much more pragmatic and performance oriented. This approach is going to avoid the "snowflake" effect regarding the decrease in performance that is usually notable as the volume of data increases. In this work, we introduced a new process mining based technique that, using the detailed execution records of an ETL process – the so-called logs, allows us to discover the nature of the ETL process behind these logs. In possession of detailed data concerning each step of our process (mean time, absolute frequency, etc), we can define a new mathematical model that illustrates the "well-being", that is, the degree of performance of our system, by establishing the correlation between the collected variables. Thus, by making it accessible to the system admins, we're introducing a new paradigm regarding the development and maintenance of ETL processes, much more concerned with issues like the performance or the knowledge behind the impact of our architectural decisions, mainly when we're deciding about the components we're going to use to execute each step of our ETL.

Índice

Índice de Figuras

Índice de Tabelas

Capítulo 1

Introdução

1.1 Contextualização

Os processos ETL – *Extract-Transform-Load* – desempenham um papel "chave" nos sistemas de *Data Warehousing*. Com efeito, o facto de serem os responsáveis pelo povoamento de um *Data Warehouse* é, por si só, uma tarefa de grande relevância, à qual acresce toda a dificuldade associada à gestão do período de disponibilidade do sistema operacional para obtenção dos dados, à conciliação dos dados provenientes de diversas fontes, bem como à gestão de um elevado volume de dados que podem conter todo o tipo de inconsistências e omissões que ameaçam o processo de suporte à decisão, o qual é o mote para a construção deste tipo de sistemas.

A simples descrição de um sistema ETL não ilustra, de todo, a complexidade da peça de *software* na sua génese. Parte dessa complexidade e dificuldade inicial deve-se, muitas vezes, à inexistência de um processo sistemático e linear de implementação que permita orientar minimamente as linhas gerais da arquitetura do sistema. Assim, e tendo em conta este cenário, a implementação do processo acaba por ser realizada de forma *ad-hoc*, conduzindo a um produto final funcional, mas pouco fiável, no sentido em que não se conseguem prever todas as exceções que podem ocorrer quando se lida com o elevado volume de dados de entrada - *input* - a cada ciclo de execução do sistema. Ao não antever todas as exceções possíveis, mais tarde ou mais cedo, poderão ocorrer períodos de *downtime* no sistema e, consequentemente, situações em que a infraestrutura poderá não traduzir os dados mais recentes, podendo precipitar algum processo de tomada de decisão

que, assim, não terá em consideração os últimos dados do sistema operacional. O "trabalho" a executar por um sistema de ETL está diretamente dependente da forma como as fontes de dados lidam com os dados e asseguram a qualidade dos mesmos. Nesse sentido, por diversas vezes, a elevada permeabilidade dos sistemas operacionais não se faz notar até ao momento em que se pretende visualizar os dados de uma outra forma, através de estruturas como os *Data Warehouse*, por exemplo. Numa situação ideal estes sistemas não deveriam ignorar dados incorretamente formados, nem entrar em período de *downtime,* por não serem capazes de lidar com situações de exceção. Deveriam, sim, implementar tabelas de quarentena na área de retenção que identificassem os registos e apontassem um potencial problema, de forma a poder ser proposta uma resolução mais rápida pelos arquitetos do sistema.

A imprevisibilidade dos dados, aliada a uma implementação pouco consistente, faz com que se torne natural concluir que, mediante certas condições, um sistema ETL possa demorar mais a executar, quando comparado com outras situações. É precisamente essa imprevisibilidade, que faz dos sistemas ETL peças de *software* bastante sofisticadas, mas, ao mesmo tempo, frágeis. Ora, o tempo de execução é porventura o aspeto mais crítico em sistemas ETL, uma vez que estes processos estão geralmente circunscritos a uma janela de oportunidade relativamente reduzida. Daí, a necessidade de se introduzirem otimizações no processo ETL e de se tornar o seu desempenho o mais independente possível dos dados extraídos dos sistemas operacionais.

Para se detetar, caracterizar e minimizar potenciais pontos negros de um sistema ETL, precisa-se de se ter um processo ETL bem conhecido como base, bem como as *logs* com o registo detalhado da sua atividade ao longo do tempo. Depois disso, tendo como base de trabalho o conteúdo dessas *logs* e um conjunto de técnicas de *process mining*, será possível obter elementos bastante ilustrativos da orgânica do processo, que fornecerão informação bastante pertinente sobre os diversos ciclos de execução do sistema ETL. Depois, consoante os dados resultantes da mineração de dados, poder-se-á desenvolver (ou melhorar) um *Data Warehouse* que os acolherá, sobre o qual se implementará uma estrutura multidimensional que permitirá traçar um perfil temporal de execução do ETL, mediante o cálculo de um índice de bem-estar, de acordo com um conjunto de parâmetros de funcionamento previamente definidos. Esse índice permitirá identificar situações anómalas de execução de forma expedita e verificar, em detalhe, o que as originou, uma vez que é calculado tendo por base o cubo referido anteriormente, que contém todos os dados resultantes da mineração do processo.

1.2 Motivação

Depois de desenvolvida a estrutura do sistema ETL, é usual passar-se à sua fase de *deployment*, sem qualquer tipo de reservas. Porém, como seria expectável, o desempenho do sistema acaba por tornar-se irregular. Naturalmente que quando a principal preocupação que preside ao desenvolvimento destes sistemas assenta na obtenção de um protótipo minimamente funcional ao qual vão sendo acrescentados mais detalhes ao longo do ciclo evolutivo, o resultado final acaba por não ser satisfatório. Este cenário tende a piorar quando os testes que são efetuados numa etapa prévia à entrada em produção do sistema não refletem a carga real e sobretudo o tipo de *input* a que este irá estar sujeito. É precisamente quando ocorre a entrada em produção que tende a tornar-se claro que a abordagem seguida foi claramente deficitária e que foram marginalizadas as etapas de testes e a sua relevância, no sentido de verificar se o sistema é resiliente e eficaz face à imprevisibilidade dos dados a que irá estar sujeito e à forma como os irá filtrar antes de os acolher na infraestrutura final. Logicamente que a preocupação com a resiliência e eficácia do *data warehouse* não incide apenas na capacidade desta infraestrutura apresentar os dados necessários ao suporte à decisão, mas também na capacidade que este mecanismo tem de disponibilizar os dados em tempo útil, uma vez que esse é por definição um dos grandes motes à implementação deste tipo de sistemas. Perante estas circunstâncias, a disposição de quem desenvolve este tipo de sistemas altera-se, tendendo a inclinar-se para aquela que deveria ter sido a postura inicial. Assim sendo, a equipa de desenvolvimento apercebe-se da importância de perceber bem o tipo de dados com que está a lidar (etapa de *profilling*) para, a partir de aí delinear uma estratégia que permita ao futuro *data warehouse* assumir a função que dele se espera, que é a de traduzir os factos e disponibilizar os dados necessários ao processo de suporte à decisão em tempo útil. Ou seja, daqui percebemos a importância de aliar um elevado grau de conhecimento dos dados com que estamos a lidar à questão da *performance*, não descurando a sua relevância no sentido de tornar o sistema realmente útil. Prosseguindo a análise anterior, as preocupações com o desempenho destes processos surgem fundamentalmente motivadas por períodos de indisponibilidade destes sistemas, ou até mesmo, por situações de incumprimento da janela de oportunidade, com todas as complicações inerentes. Precisamente quando confrontados com o ambiente real, os sistemas ETL tendem a comportar-se de forma irregular consoante o tipo de *input* a que estão sujeitos, surgindo pontos de estrangulamento na *performance* que levam a um atraso de todas as etapas subsequentes. Os piores casos são aqueles em que efeito desses pontos tende a piorar com o aumento do volume de entrada dos dados. Estes pontos, por nós designados como pontos negros, são, assim, o ponto de partida deste trabalho de dissertação, o qual se centra na sua descoberta,

caracterização e possível resolução. Além disso, em sistemas de ETL o tempo é um fator crítico, pelo que qualquer aspeto que venha a degradar a sua *performance,* e hipoteticamente estreitar ainda mais a sua janela de oportunidade, deverá ser mitigado a fim de não comprometer a janela temporal definida e a funcionalidade expectável para este tipo de sistemas. Mais do que analisar a temática do desempenho de processos ETL, de um ponto de vista particular, neste trabalho aborda-se, também, o aspeto da "universalidade" deste tipo de sistemas e da influência que estes têm no quotidiano de uma organização.

1.3 Objetivos

No âmbito deste projeto de dissertação estabeleceu-se um objetivo principal: definir um método de deteção e resolução de pontos negros em sistemas ETL que permitisse, na fase de instalação e arranque do sistema, verificar se o seu desempenho se adequa aos parâmetros de funcionamento e aos requisitos operacionais estabelecidos. Neste trabalho, sugere-se que a verificação do desempenho de um sistema de ETL seja realizada através de um índice de adequação ao processo implementado num dado contexto de aplicação. Nesse sentido, dever-se-á idealizar e implementar um método que permita correlacionar os aspetos mais relevantes no condicionamento do desempenho de um processo de ETL (e.g. volume de dados, janela de oportunidade, tempo de execução, ou a energia consumida) e, dessa forma, seja capaz de avaliar o nível da sua adequação. Assim sendo, pretendemos monitorizar um sistema de ETL de uma forma o mais rigorosa possível, mas fazendo com que isso não implique qualquer complexidade extra para os arquitetos deste tipo de sistemas. Desta forma, consegue-se avaliar o "bem-estar" do sistema de ETL face à sua condição atual e, com isso, assegurar um processo de monitorização permanente de forma a avaliar, em tempo considerado útil, eventuais anomalias do sistema indicadas por uma descida gradual ou abrupta do índice que anteriormente referimos.

A avaliação do bem-estar de um sistema de ETL torna-se interessante e importante para ciclos de execução que estejam a ser realizados correntemente. Porém, seria uma enorme mais valia se, de alguma forma, os administradores de sistemas de ETL, responsáveis pela sua gestão e operacionalidade, pudessem antecipar o desempenho de execuções futuras, bem como o seu comportamento, com base no conhecimento da evolução do índice de bem-estar implementado, da sua história e valores de suporte. Tal conhecimento, permitiria aos administradores de sistemas de ETL conceberem planos de contingência de forma mais atempada e ajustada às condições operacionais do momento, contornando assim os sempre indesejáveis períodos de *downtime* do

sistema. Complementarmente, com este trabalho também pretendemos disponibilizar um sistema de *dashboarding,* preferencialmente interativo, que permita verificar, a cada momento, o estado do ETL, bem como avaliar a evolução dos vários pontos negros detetados ao longo do tempo (Figura 1).

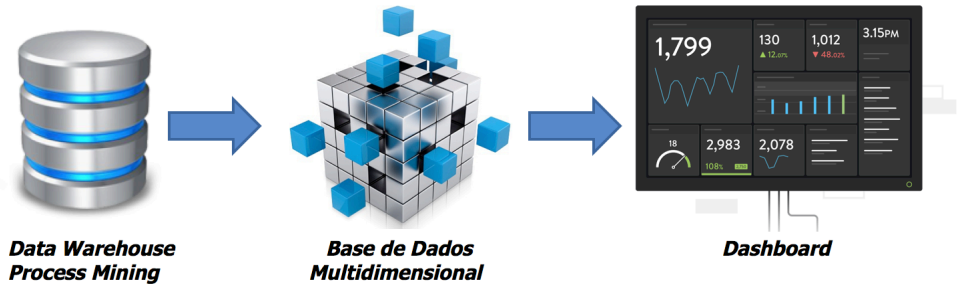


Figura 1 – Esquema ilustrativo da sequência dos processos que foram implementados.

1.4 Organização do Documento

Para além do presente capítulo, esta dissertação integra três outros capítulos, que apresentam, respetivamente:

Capítulo 2 – Identificação de Pontos Negros – neste capítulo relatamos o processo de identificação de pontos negros, desde uma contextualização de *Process Mining* e das suas reais mais valias na descoberta de processos, passando pela apresentação de uma sequência lógica de passos para deteção de pontos de estrangulamento, pela seleção e caracterização de um sistema ETL para análise, pela obtenção de *logs* desse processo, pela seleção de uma ferramenta que aplique técnicas de *process mining* a essas *logs* e terminando com a deteção de potenciais pontos negros.

Capítulo 3 – Geração de Perfis de Execução ETL – nesta parte do documento abordamos a forma como de forma sistemática iremos ser capazes de gerar perfis de execução de sistemas ETL. Nesse sentido, começamos por mostrar que a análise a efetuar às redes de execução obtidas dos sistemas terá de ser mais do que meramente empírica. Posteriormente, refletimos sobre a necessidade de enriquecer os registos de execução obtidos, no mínimo com a data e hora associados à sua execução, entre outros atributos relevantes. Seguidamente e ainda neste capítulo, apresentamos um esquema dimensional

para o *data warehouse* que irá ser implementado, bem como o processo ETL que manterá a infraestrutura atual. Terminamos com a apresentação do modelo da base de dados multidimensional que será implementada no sentido de agilizar todo o processo de análise das métricas e indicadores recolhidos.

Capítulo 4 – Um Índice de bem-estar para ETL – neste capítulo da dissertação, desenvolvemos a expressão de cálculo do índice de bem-estar, bem como os processos de previsão do índice e de pontos negros e definição das respetivas infraestruturas – *data warehouse* e base de dados multidimensional. Concluímos este capítulo com a apresentação do *dashboard* final onde todos estes indicadores se tornam visíveis.

Capítulo 5 – Conclusões e trabalho futuro – neste capítulo apresentamos algumas conclusões sobre este trabalho de dissertação, analisando aspetos positivos e negativos que o tenham envolvido. Concluímos o capítulo com alguma análise relativa ao futuro do método desenvolvido e da sua aplicação em ambientes reais.

Capítulo 2

Identificação de Pontos Negros

2.1 Mineração de Processos

A utilização de *process mining* em organizações com vista a melhor perceber a orgânica dos processos operacionais (e melhorá-los) tem vindo a aumentar. Contudo, estas técnicas possuem, ainda, algumas limitações que deverão ser acauteladas. Os modelos descobertos a partir de técnicas de mineração de processos não devem ser encarados como totalmente representativos da realidade que pretendem modelar. Com efeito, e depois de se descobrir um modelo real para representação de um processo, existem inúmeras técnicas de *process mining* para, por exemplo, analisar pontos de estrangulamento no desempenho de sistemas (*bottlenecks*), descobrir ineficiências, verificar a conformidade, explicar desvios, prever *performance* e orientar os utilizadores em direção a melhores processos. Essas técnicas deverão ser tidas em conta se se pretende melhorar os resultados da aplicação de *process mining* [1].

No domínio do *process mining*, as *logs* de eventos são a principal "matéria-prima", sem a qual o processo de mineração não poderá avançar. O facto de serem um recurso crítico não faz com que essas *logs* sejam fáceis de obter, pelo contrário. Assim, muitas vezes, a disponibilidade dos registos de execução de um dado processo é um problema. Outras vezes esses dados contêm omissões ou precisam de ser enriquecidos para se tornarem aptos a retratar a essência do processo na sua origem. Para além disso, muitas técnicas não avaliam a forma como a *log* de

eventos está definida, não se assegurando, por exemplo, que os eventos estejam bem definidos, i.e., que se referem a exatamente um caso e uma atividade que foi realizada de alguma forma. Existem, contudo, ferramentas para a geração automática de *logs* de eventos que podem facilitar o processo de obtenção, no entanto, muitas delas não registam os eventos explicitamente [1].

A aplicação de *process mining* a uma *log* está muitas vezes dependente da escolha da ferramenta de mineração. Porém, outras dificuldades emergem após ter sido realizado o processo de escolha. Assim, estabelecer a "ponte" entre os sistemas de informação e as ferramentas de *process mining* está longe de ser trivial, revelando-se a instalação e manutenção destas infraestruturas bastante custosa, sendo o mapeamento das informações recolhidas na ferramenta apenas um exemplo de uma tarefa vital no âmbito da mineração de processos [2]. As tradicionais técnicas de *process mining* têm grande dificuldade em lidar com processos não estruturados, pelo que se torna importante determinar os casos principais e separá-los dos casos desviantes. Essa separação é deveras importante, uma vez que alguns casos ocorrem apenas uma vez e ao serem considerados no processo de análise influenciam negativamente o processo de obtenção do modelo final, o que se traduz posteriormente em resultados potencialmente enganosos [3]. Geralmente, os ambientes mais imprevisíveis estão, também, associados a uma complexidade de análise acrescida. Uma solução possível para ultrapassar tal circunstância passa pela separação dos dados disponíveis em vários *clusters*, contendo subconjuntos homogéneos, e, para cada subconjunto, criar-se um modelo de processo autónomo. Desta forma, os resultados da aplicação de *process mining* em ambientes flexíveis tendem a melhorar [4].

Contudo, o *process mining* não resume a totalidade das técnicas de mineração existentes. A área de mineração de dados divide-se em várias valências, dependendo do problema a tratar. Veja-se, por exemplo, o caso de *pattern mining* que incide, sobretudo, na descoberta de padrões estatisticamente relevantes em processos cujos eventos ocorrem de forma parcialmente ordenada. Ora, essa área de mineração não tem em conta as instâncias dos processos, uma vez que aquilo que se pretende é apresentar uma nova técnica e a sua correspondente implementação, que descobre episódios frequentes numa *log* de eventos, tirando partido do facto dos eventos estarem associados a casos. Além de descobrir padrões em processos complexos e de avaliar a sua conformidade, baseando-se numa ordenação parcial dos eventos, também descobre regras de previsão de comportamento e de comportamentos correlacionados, bem como aplica a técnica a outras perspetivas presentes em *logs* de eventos, o que poderá ser, de igual forma, um bom ponto

de partida para análise de pontos negros em processos ETL [5]. Por fim, refira-se que, a necessidade de avaliar as técnicas de *process mining*, quanto à sua real aplicação, tem vindo a ganhar força entre a sua comunidade de utilizadores. Atualmente, não há qualquer tipo de preocupação em avaliar a qualidade dos modelos descobertos pelos algoritmos de *process mining*, pelo que uma qualquer *framework* de avaliação que permita aos investigadores da área comparar a *performance* dos seus algoritmos e aos utilizadores finais validar os seus resultados, será algo extremamente útil [6]. De seguida, em termos gerais, abordaremos o processo de deteção de pontos negros e, em particular, a forma como nele aplicámos algumas técnicas de *process mining.*

2.2 O Processo de Deteção

A visão *ad-hoc* que atualmente vigora na conceção e implementação de sistemas ETL não assegura a robustez e o nível de desempenho necessários que processos desta natureza usualmente requerem. Com efeito, este tipo de processos desempenha uma função crítica no âmbito dos sistemas de *data warehousing*, uma vez que são os responsáveis pela manutenção e povoamento de todas as estruturas de dados de um *data warehouse*, com os mais recentes dados provenientes dos seus sistemas operacionais. Nesse sentido, e no sentido de alcançar o objetivo principal definido para este projeto de dissertação, desenvolveu-se um método específico para a deteção e resolução de pontos negros de ETL (Figura 2), que irá sendo detalhado ao longo dos próximos capítulos desta dissertação. Esse método, constituído por cinco fases, contribui decisivamente para uma maior robustez dos processos implementados, ao permitir detetar situações de potencial quebra de *performance*.

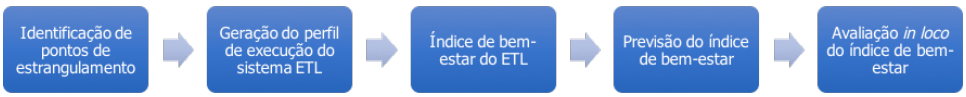


Figura 2 – Metodologia desenvolvida para deteção e resolução de pontos negros.

Vejamos, então, cada uma das etapas apresentadas na Figura 2. A primeira etapa diz respeito à definição de um subprocesso para identificação de pontos de quebra na *performance* do ETL que inclui: a seleção de um sistema ETL exemplo, a obtenção automática de *logs,* detalhadas a cada ciclo de execução, e a escolha de uma ferramenta de *process mining* para descoberta do processo. Na segunda etapa, pretende-se mostrar como se generaliza a etapa anterior e se tira partido das redes de execução geradas pela ferramenta escolhida, extraindo-as para um formato manipulável

e introduzindo as infraestruturas adequadas – o *data warehouse* e a base de dados multidimensional - a uma análise do ponto de vista temporal do bem-estar do ETL. Na terceira etapa, apresenta-se a expressão matemática definida para cálculo do índice de bem-estar do ETL, esclarecendo a origem de cada um dos parâmetros da função. Na quarta etapa do processo, introduz-se o conceito de previsão e a sua necessidade no âmbito da monitorização de sistemas ETL. Com efeito, sendo o objetivo primordial deste projeto de dissertação assegurar um nível de qualidade de serviço elevado, introduzir a previsão de pontos negros, bem como o respetivo índice de bem-estar, é um passo natural no desenvolvimento do processo estabelecido. Para concluir, na quinta etapa, introduz-se uma alternativa para visualização dos dados disponibilizados pelo sistema de monitorização, nomeadamente plataformas de *dashboarding* [7] que garantem uma elevada acessibilidade para as várias partes interessadas, ao mesmo tempo que asseguram uma visão em tempo real dos indicadores chave, neste caso o índice de bem-estar atual e os pontos negros detetados.

Como vimos, a identificação de pontos de estrangulamento na execução de um sistema ETL é a primeira etapa do método anteriormente apresentado. Sendo o método apresentado um fluxo contínuo, esta etapa revela-se crítica para o sucesso das etapas posteriores. Apesar disso, o título escolhido não permite antever o conjunto e a complexidade das sub tarefas que deverão ser executadas nesta altura do processo. Assim, é necessário selecionar um sistema ETL para análise e caracterizá-lo devidamente. Seguidamente é necessário averiguar se o sistema selecionado já tem algum mecanismo de geração automática de *logs* e se este é suficiente para, mais tarde, se reconstituir o processo através de uma ferramenta de *process mining*. Caso não seja, poderá ser necessário realizar algum trabalho extra para fazer a definição de um registo completo de eventos. Volvida essa fase, aplicam-se as *logs* obtidas na ferramenta de *process mining* selecionada, sendo depois reconstituído o ciclo de execução e identificados potenciais pontos de quebra na *performance* do sistema.

2.3 Seleção e Caracterização de um Sistema ETL

Para verificar a presença de pontos de estrangulamento na *performance* de um *package* ETL, é preciso saber qual o processo que se pretende monitorizar. Para isso, selecionámos um processo ETL que foi implementado para povoar um *data warehouse* que concilia a atividade de negócio de um *cluster* composto por três empresas [8]. Elaborado no contexto de uma unidade curricular do

perfil de *Business Intelligence*, do Mestrado em Informática da Universidade do Minho, este *package* ETL concilia, diariamente, dados de três empresas (fictícias) distintas:

1. *Tuk Tours*, uma empresa de aluguer de *tuk tuk* na cidade de Lisboa para passeios turísticos;
2. *Make It Yours*, uma empresa de *rent-a-car* com sede em Portugal;
3. *JustMovies*, uma empresa norte americana de aluguer de filmes.

O facto de serem empresas sediadas em continentes distintos, faz com que a janela de oportunidade para execução do processo de ETL esteja, logo à partida, condicionada, implicando uma grande disciplina no cumprimento do intervalo de tempo definido para o efeito. Assim, e para que o povoamento do *data warehouse* ocorra a tempo da abertura das lojas em Portugal, por exemplo, e se tenha em consideração a atividade comercial em território norte americano, a janela de oportunidade fica circunscrita a um intervalo de apenas 2h (Figura 3).

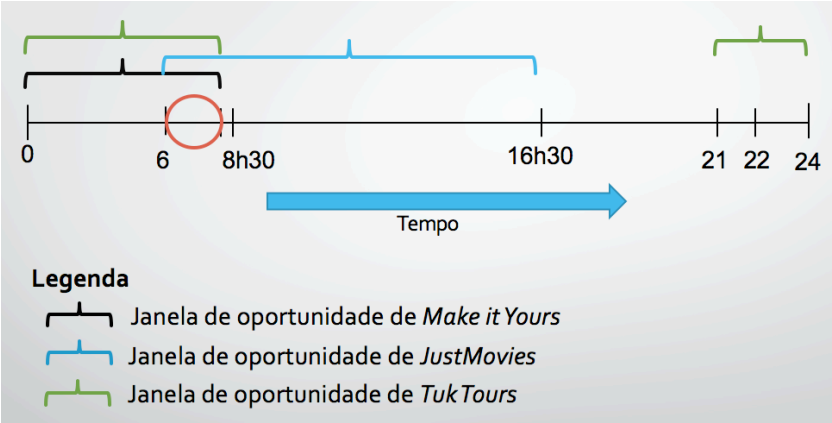


Figura 3 – Sobreposição das várias janelas de oportunidade e janela de oportunidade final para o sistema ETL.

O processo de povoamento, propriamente dito, é executado diariamente entre as 6h e as 8h (fuso horário português), com a invocação de um *package* de software implementado em *Kettle* [9]. As tarefas de ETL que são executadas ao longo de todo o processo não são, de todo, idênticas para as três empresas. Com efeito, tanto para a empresa *Make It Yours* como para a empresa *JustMovies*, as tarefas de ETL recorrem a sistemas operacionais assentes em bases de dados relacionais. Para estes casos a política de extração de dados adotada implicou o desenvolvimento e instalação de um conjunto específico de *triggers* nos sistemas operacionais. Os *triggers* implementados reagem a modificações ocorridas nos objetos das bases de dados sobre os quais

estão definidos, colocando em tabelas de auditoria um registo das modificações ocorridas. Todavia, no que diz respeito à empresa *Tuk Tours*, os dados do sistema operacional estão organizados em folhas de cálculo, com manutenção diária, cujo conteúdo é constituído apenas por novos registos de dados ou alterações a registos previamente inseridos. Assim, podemos ver que o sistema de povoamento implementado atua sobre um conjunto de fontes heterogéneas, o que aumentou, obviamente, a complexidade dos processos de conciliação envolvidos, sobretudo pelo facto de as folhas de cálculo serem, a priori, uma fonte de dados um pouco "permeável" a eventuais situações de inconsistência de dados. Quanto ao volume de dados envolvido diariamente no processo (Tabela 1), este foi, também, estimado e, uma vez mais, as diferenças encontradas entre os valores apurados corroboram a natureza profundamente distinta das três empresas envolvidas no processo de povoamento.

Tabela 1 – Estimativa do volume dados envolvidos num processo ETL diário.

Fonte de dados	Clientes novos/dia	Alugueres/ dia	Entrada *Stock* ano[1]
Make it Yours	10	15	5
JustMovies	30	150	45
Tuk Tours	10	10	N/D

2.4 As *logs* de Eventos

O *package* ETL que selecionámos como instrumento para o nosso trabalho foi implementado em *Kettle.* Apesar desta ferramenta possibilitar a obtenção de *logs* de processo, bem como o seu registo de forma automatizada numa tabela relacional, estas não possuem o nível de detalhe necessário à análise do processo que se pretende fazer. Assim, foi necessário implementar um mecanismo alternativo de *logging* que passou pela injeção manual de etiquetas temporais que indicassem, o início e término de cada tarefa ETL, bem como outros atributos relevantes relacionados com o funcionamento do sistema de ETL. Essa injeção manual dos registos numa tabela relacional atuando como uma log sequencial, implicou a utilização do componente "Execute SQL statements" [10] antes de cada passo de cada transformação, por forma a se poder identificar aquilo que estava a ser executado e quando estava a ser executado. Apesar disso, convém não esquecer que *Kettle*, como uma ferramenta destinada à implementação de processos ETL,

[1] O "stock" referido corresponde a novos automóveis e novos filmes para aluguer.

paraleliza ao máximo as várias tarefas que compõem um determinado package, dando assim primazia a todos os aspetos diretamente relacionados com *performance*. Nesse sentido, e não havendo qualquer tipo de dependência de passos anteriores, esta ferramenta executa vários passos (tarefas) em simultâneo, o que significa que a inserção dos registos na *log* reflete este tipo de execução. Prevalecendo esta característica do *Kettle*, a *log* obtida terá o registo dos vários passos executados, mas os instantes temporais medidos serão praticamente idênticos, o que, obviamente, não traduz aquilo que, de facto, aconteceu no processo. De forma a contornar esta situação utilizou-se o componente "Block this step until steps finish" [11]. Este componente permite bloquear a execução de um dado passo até que um ou mais passos indicados, consoante o necessário, tenham concluído. Assim, conseguimos aguardar pelo início efetivo de um determinado passo antes de ser inserir o registo na tabela relacional implementada na área de retenção do *data warehouse* (Figura 4). De referir, porém, que, por uma questão de simplificação, o tempo que decorre entre o término de um passo e o início do passo seguinte foi considerado como irrelevante, o que permitiu considerar que o instante de tempo final de um dado passo é idêntico ao instante de tempo inicial do passo seguinte.

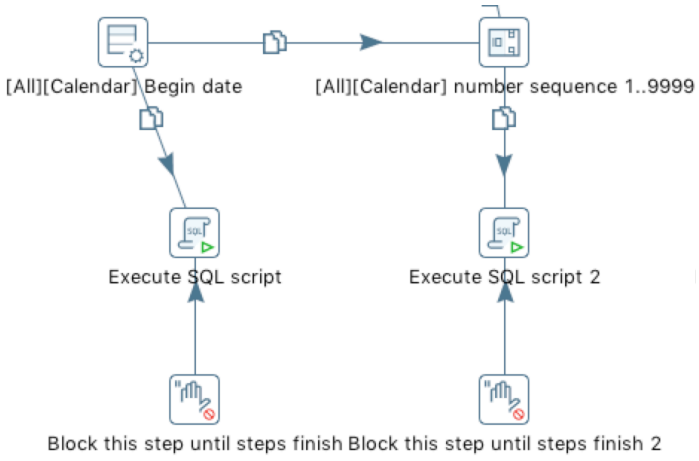


Figura 4 – Extrato de uma transformação *Kettle* alterada para obtenção das *logs* de execução do processo ETL.

De acordo com a estratégia implementada, no momento em que o processo ETL termina é possível extrair o registo de execução do sistema respeitante a esse ciclo e importá-lo a partir da ferramenta de *process mining.* De seguida, esta ferramenta poderá fazer a reconstituição do processo, traçando a rede de execução e calculando as diversas métricas consideradas, como a média e a mediana do tempo, a frequência absoluta, entre outras. Para que isso seja realizado,

bastará filtrar a *log* pelo valor do atributo "*executionOrder*" atual, valor este que é incrementado sempre que o package ETL é invocado. Este atributo permite identificar o número da execução atual.

logId	stepname	componenteKettle	beginTime	endTime	executionOrder	trace
1	Start!	Execute SQL statements	2016-03-26 21:53:31.050400	2016-03-26 21:53:31.050400	1	Start
2	[Sakila][Customer] Select registers	Select / Rename values	2016-03-26 21:53:31.050400	2016-03-26 21:53:31.256100	1	Customer SK
3	[Sakila][Customer] Verify ER	Filter rows	2016-03-26 21:53:31.256100	2016-03-26 21:53:31.269700	1	Customer SK
4	[Sakila][Customer] Operation type	Filter rows	2016-03-26 21:53:31.269700	2016-03-26 21:53:31.271100	1	Customer SK
5	[Sakila][Customer] DimclienteHist 2	Table output	2016-03-26 21:53:31.271100	2016-03-26 21:53:31.273200	1	Customer SK
6	[Sakila][Customer] Delete AudDimCliente	Execute SQL statements	2016-03-26 21:53:31.273200	2016-03-26 21:53:31.274000	1	Customer SK

Figura 5 – Excerto da *log* de eventos obtida através do mecanismo de *logging* desenvolvido.

No que diz respeito à estrutura do registo de execução recolhido (Figura 5), esta é constituída por um conjunto de atributos que caracterizam a ocorrência de um dado evento, acolhendo a identificação do componente *Kettle* (*componenteKettle*), utilizado num dado passo (*stepname*), de uma determinada transformação (*trace*) que ocorreu entre os instantes de tempo *beginTime* e *endTime*, num determinado ciclo de execução (*executionOrder*). De seguida analisaremos a seleção da ferramenta de *process mining* que utilizámos para reconstituir o processo ETL e, consequentemente avaliar o seu desempenho.

2.5 A Ferramenta de *Process Mining*

A mineração de processos é uma área que nos últimos anos teve grande expansão, dadas as suas potencialidades para a descoberta de ineficiências em processos de natureza muito diversa. Hoje, no mercado, já existem algumas ferramentas de *process mining* com capacidade para traçarem redes de execução de processos, de forma bastante efetiva, recorrendo a algoritmos bem conhecidos. Porém, o que acontece com a maioria destas soluções é que a sua *interface* e usabilidade é bastante descurada, o que complica bastante a tarefa dos utilizadores menos experientes. Inicialmente, julgou-se que a ferramenta *ProM* [12] seria o *freeware* mais adequado para traçar redes de execução de processos. No entanto, rapidamente se percebeu que a sua *interface* (Figura 6), de baixo nível, iria complicar em muito a obtenção das próprias redes de execução, para que estas fossem legíveis e interpretáveis, especialmente em situações em que são trabalhadas algumas centenas de instâncias de *log* produzidas a cada execução do sistema ETL. Além disso, a *interface* disponibilizada assenta muito na listagem dos vários algoritmos que temos ao dispor e na configuração de alguns parâmetros, com características bastante avançadas. Isto exige um nível de conhecimento bastante profundo por parte dos seus utilizadores. Foi precisamente por estes motivos que, na procura por uma alternativa, se encontrou a ferramenta

Disco [13]. Esta ferramenta, apesar de não ser *Open Source,* permite um licenciamento para a comunidade académica. Além disso, esta ferramenta, torna possível, de uma forma muito intuitiva e em poucos segundos, obter a rede de execução de um processo de forma clara, desde que se importem os registos do processo e se mapeiem os vários atributos na aplicação (Figura 7). Além disso, permite sobre a rede de execução obter vários tipos de vistas para análise, quer seja ao nível da frequência de ocorrência de cada um dos elementos, quer seja ao nível temporal, utilizando uma escala de cores bastante intuitiva e permitindo a identificação de valores atípicos de uma forma praticamente imediata. Apesar de ser de fácil obtenção, essa identificação requer um conjunto de considerações prévias, uma vez que a definição de ponto negro não é lata ao ponto de se considerar um ponto negro um qualquer componente do processo de ETL cujo tempo de execução tenha excedido um determinado limite predefinido. Todas estas considerações serão esclarecidas posteriormente nesta dissertação.

Figura 6 – *Interface* gráfica da ferramenta de *process mining*, *ProM.*

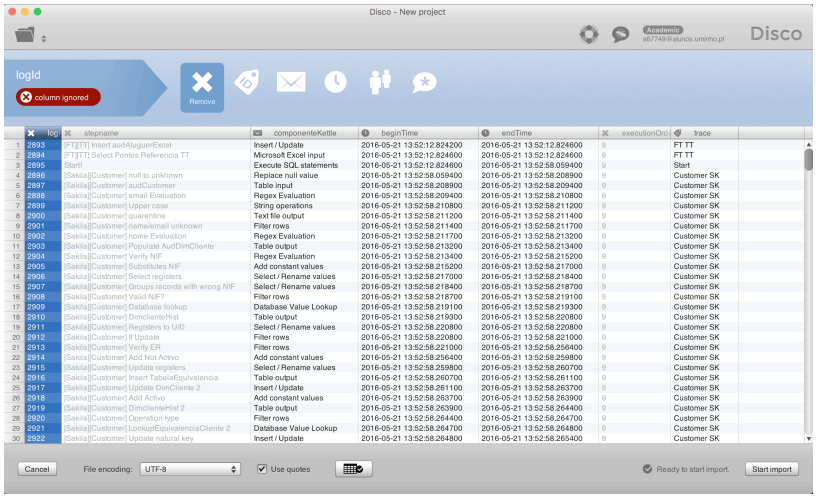


Figura 7 – *Interface* gráfica da ferramenta de *process mining*, *Disco*.

2.6 A Deteção de Pontos Negros

Uma vez obtidos os registos de execução do processo ETL e selecionada a ferramenta de *process mining* mais adequada, iniciou-se o processo de análise das logs através da ferramenta *Disco*, começando-se por fazer mapeamento de cada um dos atributos envolvidos, um a um, para traçar a rede de execução. Esse mapeamento, relatado em [14], para obtenção da rede de execução de um ciclo de ETL e cálculo das respetivas métricas associadas, envolveu, para cada atributo integrante da tabela *log*, a atribuição de um significado. Assim atribuímos:

- "*Case*" para o atributo "*trace*", ou seja, para o atributo que designa o nome genérico da transformação a que o evento registado está associado;
- "*Activity*" para o atributo "componenteKettle", ou seja, o atributo que define efetivamente a atividade que desencadeou o evento;
- "*Timestamp*" para os atributos associados aos instantes temporais de início, "*beginTime*", e término de um determinado evento, "*endTime*";
- "*Remove*" para os atributos irrelevantes à nossa análise, no caso, "*logId*" que representa a chave primária utilizada na implementação da tabela *log*, "*stepname"* que apesar de relevante é um atributo que não permite uma análise tão refinada quanto a que pretendíamos e "*executionOrder*", atributo que apenas permite distinguir os vários dias de execução do nosso processo ETL, sendo irrelevante para a rede a traçar;

- "*Resource*" é utilizado para traçar *logs* que tipicamente tenham associado a um evento um recurso, seja ele um trabalhador, uma impressora, etc. Neste caso não temos qualquer atributo dessa natureza.

Terminado este processo e tendo em conta o mapeamento apresentado, traçou-se a respetiva rede de execução (Figura 8).

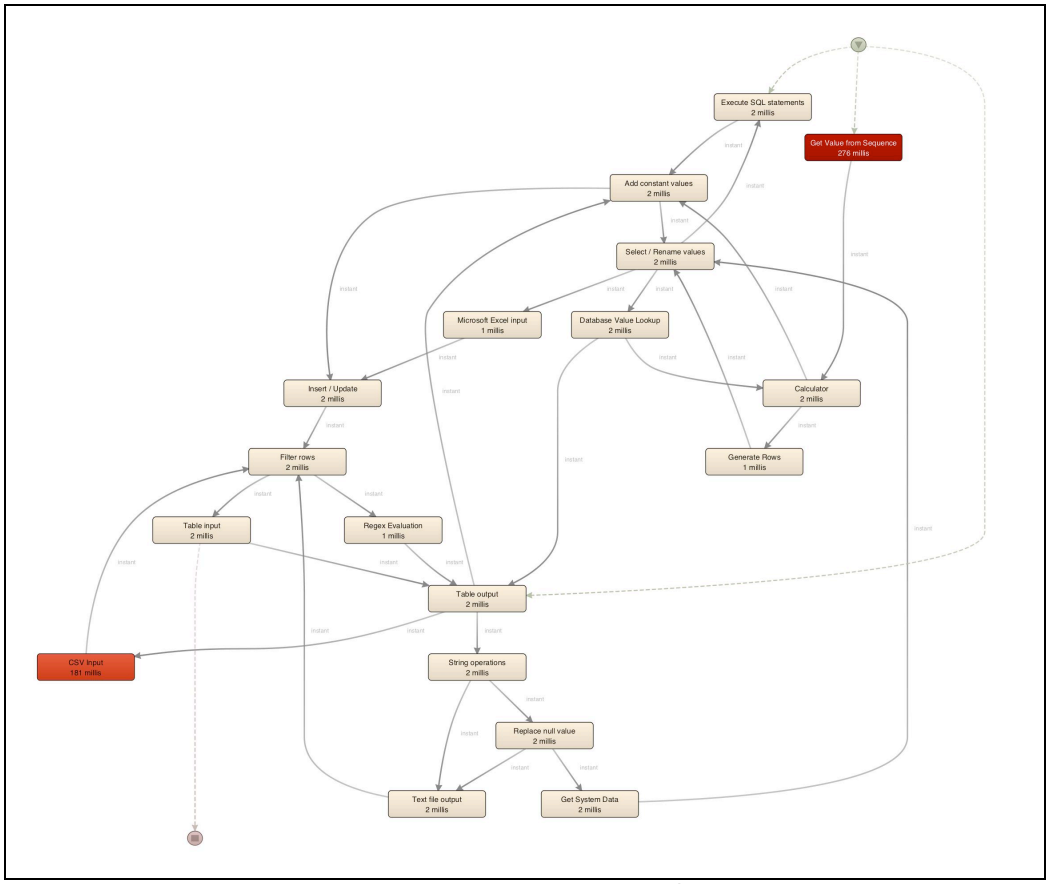


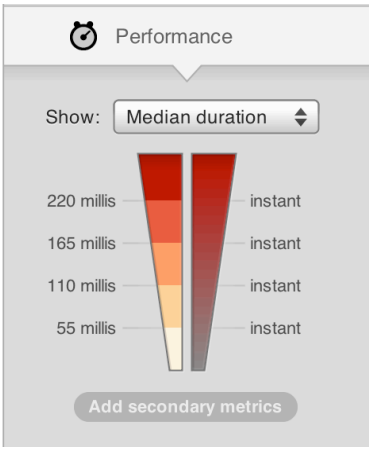
Figura 8 – Exemplo de uma rede gerada pela ferramenta *Disco*.

Como já referido, as *logs* de execução dos processos ETL são essenciais para a realização da análise que pretendemos fazer. Para que as obtivéssemos, a cada ciclo de execução do processo ETL, fizemos a sua geração e extraímos os dados que necessitávamos. Depois, mapeámos cada um dos atributos na ferramenta *Disco* e gerámos a respetiva rede de execução. Para que acedêssemos ao registo de eventos, bastou fazer apenas uma simples *query* em SQL, filtrando os dados pelo dia de execução (*executionOrder*) pretendido. Contudo, a observação da rede, tendo como base o horizonte temporal de um ciclo de execução, não permite obter grandes conclusões relativamente a possíveis pontos negros, uma vez que um ciclo de execução é insuficiente para se

poder retirar qualquer tipo de ilação. Além disso, a presença de *outliers* que possam ter contribuído para uma sobrevalorização de determinado ponto da rede não deve ser descurada. Nesse sentido, retirar conclusões é um processo que requer evidências suficientemente fortes que as sustentem, uma vez que um determinado ponto poderá até ter tido um impacto considerável no desempenho de um determinado ciclo de execução, mas isso ser apenas consequência de uma maior sobrecarga de processamento da infraestrutura computacional em que está inserido, por exemplo. Por outro lado, poderá um dado ponto ter efetivamente uma *performance* comprometedora, mas o seu impacto ser reduzido por fazer parte, por exemplo, de uma dimensão sem variação. Tudo isto são hipóteses, mas que ilustram bem a necessidade de aumentar a dimensão da nossa amostra para melhorar o nosso conhecimento do processo.

Convém, também, salientar que o gradiente de cores atribuído pela ferramenta aos vários componentes da rede de execução é ajustado sempre que um novo conjunto de dados é mapeado, ou seja, é independente de todas as outras redes que a tenham precedido. Isso pode fazer com que obtenhamos um indicador um pouco enganoso, que apenas funciona de um ponto de vista visual. Se não, vejamos na Figura 9 os indicadores obtidos para os dias 1 e 3. Os indicadores apresentados para cada um desses dias correspondem a redes de execução geradas a partir do sistema ETL para o nosso caso de estudo. No entanto, basta uma rápida observação para se concluir que as diferenças não podiam ser mais claras. Assim, tendo em conta aspetos meramente visuais, pode-se concluir que a identificação de pontos negros não faz sentido e como tal carece de validação. Todavia, essa identificação é importante para a compreensão do conceito de ponto negro, mas não para retirar conclusões mais alargadas.

Dia 1



Dia 3

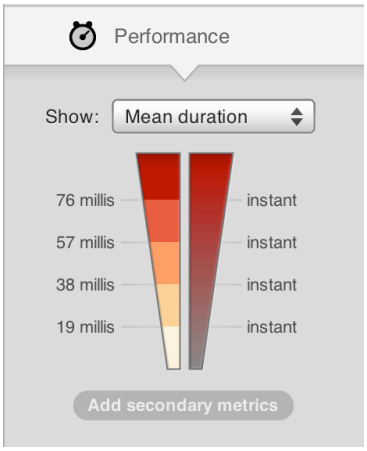


Figura 9 –Exemplo de dois indicadores de desempenho para dois dias de execução diferentes.

Capítulo 3

Geração de Perfis de Execução ETL

3.1 "Ser ou não Ser" um Ponto Negro

Previamente, definiu-se um processo para identificação de pontos de estrangulamento com base na *performance* de um *package* ETL. No entanto, esse processo não permite, por si só, avaliar a variação dos pontos negros ao longo de vários ciclos de execução do sistema de ETL. De facto, aquilo que tal processo permite fazer é uma análise sumária, pouco precisa, daqueles que são os pontos de quebra de *performance* de um *package* ETL, tendo sempre em conta o horizonte temporal do ciclo de execução ETL. Porém, convém salientar que a observação exclusiva das redes de execução de um sistema de ETL não é suficiente para que se possa considerar que os pontos assinalados pela ferramenta de análise são, de facto, pontos negros, tal como já foi revelado anteriormente. Assim, é necessário olhar ao contexto aplicacional do *package* ETL e averiguar se, de facto, o tempo de execução e, sobretudo, a frequência absoluta de utilização de um dado construtor são representativos, uma vez que para que um ponto seja de facto negro este tem que ocorrer frequentemente e, além disso, o seu tempo de execução tem que ser significativo, não bastando, pois, observar a coloração distinta apresentada pela ferramenta de análise para um determinado ponto, uma vez que a escala de cores utilizada é sempre ajustada consoante o exemplo de execução fornecido como *input*. Assim, para se conseguir retirar conclusões com algum significado é necessário definir um *threshold*, ou seja, um limite mínimo a partir do qual se considera que um determinado ponto passa a ser, de facto, negro. Esse valor, obviamente,

dependerá da experiência adquirida com a utilização do *package* ETL em questão, adquirida pelos administradores ao longo do tempo.

Depois de identificados os pontos negros e definido o processo de deteção que queremos aplicar, é necessário analisar de forma mais aprofundada cada um dos pontos que foram descobertos. Assim, para avaliar as características de cada ponto em particular, bem como as circunstâncias que conduziram ao seu aparecimento, é bastante útil que se tenha disponível um sistema de dados adequado - um *data warehouse* - complementado por um segundo sistema multidimensional, ambos orientados pelos requisitos relevantes a esta análise, de forma a que se possa cruzar as diversas perspetivas de análise de uma forma expedita e variável ao longo do tempo. Porém, estes sistemas de dados precisam de ser alimentados adequadamente, com os dados relativos aos vários ciclos de execução do ETL. Como tal, é necessário transferir as redes de execução disponibilizadas pela ferramenta *Disco* para um formato manipulável, de forma a que seja possível fazer a sua leitura e posterior integração nos sistemas de dados referidos.

A ferramenta *Disco* permite fazer a exportação das redes que gera em vários formatos. No nosso caso, o formato .csv é o mais adequado e, por conseguinte, foi o adotado para sustentar o processo de alimentação dos sistemas de dados para análise, conforme a Figura 10. Depois, para posterior manipulação, os dados exportados terão de ser analisados, no sentido de se avaliar se o seu nível de detalhe é ou não suficiente para traçar um perfil temporal sobre o comportamento do sistema ETL.

Figura 10 – *Export* da rede do processo ETL para formato *CSV*.

De forma a ter uma imagem mais clara acerca da forma como este processo se desenrola, na Figura 11, apresentamos um pequeno esquema que revela o encadeamento das várias etapas que se têm que realizar. Nessa figura identificamos quatro etapas de trabalho distintas, nomeadamente:

- Análise do Output PM – nesta etapa analisamos o *output* fornecido pelo algoritmo de *process mining* selecionado para a descoberta do processo. No caso particular deste trabalho de dissertação o algoritmo está embebido numa ferramenta, bastando apenas importar os registos de execução do processo para se obterem, de forma quase instantânea, a rede e várias métricas associadas ao ciclo de execução do processo;
- Esboço e implementação do DW – nesta etapa e depois de termos analisado o *output* resultante da aplicação de *process mining* aos registos do processo, já temos em consideração apenas as métricas que terão relevância para a análise do ponto de vista do "bem-estar" do sistema, sendo que o esboço e consequente implementação do *data warehouse* refletirão essa mesma seleção, visto que serão esses dados que irão alimentar todo o processo;
- Processo ETL para DW – uma vez implementado o *data warehouse*, nesta etapa, é necessário desenvolver todo o processo ETL para que este seja povoado com os dados relevantes;
- Base de dados multidimensional – no sentido de recorrer a ferramentas de *dashboarding*, nesta etapa do processo, passa-se à implementação de uma estrutura multidimensional suportada pelo *data warehouse* que permita a obtenção da informação relevante aos agentes de decisão, conforme definida nos requisitos de suporte à decisão na etapa de "Esboço e implementação do DW".

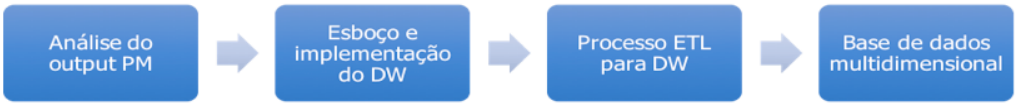


Figura 11 – Sucessão de etapas para geração de perfis de execução de sistemas ETL.

3.2 Pré-análise dos dados

Uma vez exportada a rede de execução do ETL para formato .csv, passámos à análise do *output* produzido. Nesse sentido, verificou-se que este podia ser enriquecido com alguns atributos

adicionais para suprir alguma informação básica de análise. Observando a Figura 12, facilmente se percebe que o ficheiro com a *log* não tem qualquer atributo representativo da data e da hora a que ocorreu o ciclo de execução ou quando é que este componente ETL ou aquele foi executado. Sem esses atributos, torna-se impossível traçar um perfil temporal da execução do sistema ETL. Desta forma, não é possível alcançar um dos objetivos principais deste trabalho, que foi traçado no início deste projeto de dissertação. Além disso, pode-se verificar que o nome do construtor *Kettle* (*Activity*) apesar de constar na folha de cálculo não aparece acompanhado da fase do processo (extração, transformação, limpeza) em que foi utilizado. A existência dessa informação permitiria uma análise mais interessante e detalhada do ponto de vista da avaliação do desempenho do *package* ETL, em cada uma das três grandes fases de um sistema desta natureza. Para que esse tipo de análise fosse possível, tivemos que alterar a orgânica de alguns dos componentes do sistema de ETL. Nesse sentido, o novo processo ETL terá de contemplar a angariação dessa informação de forma a que se possa fazer o enriquecimento dos dados e, consequentemente, traçar um perfil o mais detalhado possível do funcionamento do sistema. Porém, antes de se implementar o novo processo ETL para povoamento do *data warehouse*, foi necessário definir o esquema dimensional da sua base de dados, tendo em conta, precisamente, os resultados fornecidos pela ferramenta *Disco* e as condições essenciais, já relatadas anteriormente.

Activity	Absolute frequency	Case frequency	Max. repetitions	Total duration (ms)	Median duration (ms)	Mean duration (ms)	Min. duration (ms)	Max. duration (ms)
Execute SQL statements	28	10	4	2646	2	94	0	1863
Select / Rename values	46	13	6	318	2	6	1	206
Filter rows	48	14	7	886	2	18	1	287
Table output	46	18	5	1369	2	29	1	268

Figura 12 – Extrato do ficheiro *.csv* que resultou da exportação das métricas obtidas pela ferramenta *Disco*.

3.3 O Data Warehouse

Para implementar o *data warehouse*, foi necessário, em primeiro lugar, esboçar o esquema dimensional que sustentaria a sua criação. Para que este esboço fosse possível, foi necessário desenvolver um processo de levantamento de requisitos específico, que recolhesse os principais elementos de dados das estruturas dimensionais do *data warehouse*. Ora, esses requisitos surgiram, naturalmente, por análise do *output* fornecido pela ferramenta de *process mining*. Assim, de seguida, de forma resumida, apresenta-se a lista de requisitos levantados para o *data warehouse* que definimos:

1. Saber, para um determinado período de tempo, qual o tempo médio de execução de um dado construtor.
2. Saber, para um determinado ciclo de execução do ETL, o índice de bem-estar.
3. Saber, para um determinado construtor, qual a duração média do conjunto de etapas em que este foi utilizado.
4. Saber, durante um determinado período de tempo, qual o número de pontos negros registados.
5. Saber, para um determinado construtor, qual o número total de transformações distintas em que este foi utilizado.
6. Saber, para um determinado construtor, qual a duração máxima, mínima, mediana ou total que foi registada.
7. Saber para um determinado construtor, qual o número total de vezes que foi utilizado num dado contexto de um ciclo de execução.

Dos requisitos apresentados, consegue-se inferir as várias dimensões que sustentam a definição do esquema dimensional do *data warehouse*. Nesse sentido, com base nos requisitos 1, 2, 4 e 7 pudemos identificar a dimensão "Calendário". Depois, através dos requisitos 1, 3, 5, 6 e 7 surge a dimensão "Componente", uma dimensão muito útil para identificar quais os componentes *Kettle* que estiveram envolvidos na génese de um determinado ponto negro. Por fim, com a análise dos requisitos 3 e 5, obtivemos a dimensão "*CaseGrupo"*, ou seja, a dimensão que nos permite identificar o conjunto de transformações realizadas no processo ETL, mas em que um determinado componente foi utilizado pelo menos uma vez.

Tabela 2 – Matriz de decisão do *data mart Performance*.

Caracterização do *Data Mart Performance*
Identificação: *Performance*
Descrição geral: Informação para suporte à decisão na área de análise de *performance* de sistemas ETL, fornecendo dados para gestão e controlo do desempenho dos vários construtores utilizados no processo, bem como das transformações em que atuam.
Estrutura base

Tabela de factos	FT_PontoNegro
Dimensões	
Calendário	✓
CaseGrupo	✓
Componente	✓
Número de dimensões	3
Tipo	Transacional
Periodicidade	Diária
Descrição	Pontos Negros detetados ao longo de ciclos de execução do ETL
Utilidade Estratégica	Avaliação da *performance* de um sistema ETL. Deteção de pontos negros e análise dos mesmos para alterações futuras.
Utilizadores	Profissionais de BI
Observações	Nada a assinalar.

Uma vez traçada a matriz de decisão do *data mart "Performance"* (Tabela 2), foi necessário detalhar as dimensões que integram a sua estrutura. Assim, definiu-se o nível de detalhe da informação a armazenar no sistema, tendo em consideração que os pontos negros são, de facto, as entidades fulcrais em todo o problema. Nesse sentido, é natural que o grão da tabela de factos seja definido como a peça de dados mais elementar do sistema de dados, tendo a capacidade para caracterizar um ponto negro, relativo a um determinado ciclo de execução do processo ETL. O grão que foi escolhido deixa antever que, por forma a armazenar informações relevantes ao processo, como o volume de dados registado ou a proporção ocupada de janela de oportunidade do sistema, será necessário utilizar uma dimensão do tipo *junk* [15]. Desta forma, conseguimos assegurar a correção do esquema dimensional, bem como a sua coerência, face ao nível de detalhe assumido. No esquema dimensional do *data mart*, a dimensão do tipo *junk* ("*DimProcesso_junk*") está diretamente relacionada com a tabela de factos do sistema de dados ("*FT_PontoNegro*"). Na Tabela 3 podemos ver uma caracterização em maior detalhe das dimensões que constituem o *data mart "Performance"*. De salientar que a ausência da dimensão "Processo" da matriz de decisão (Tabela 2) se justifica com o facto de não derivar diretamente dos requisitos de decisão.

Tabela 3 – Caracterização das dimensões que constituem o *data mart "Performance"*.

Dimensões do data mart Performance			
Nr	Identificação	Descrição	Esquema (Tipo)
1	Calendário	Dimensão temporal que sustenta as análises ao longo do tempo, por exemplo anual, mensal, semanal, etc	DimCalendario (Normal)
2	CaseGrupo	Identificação e caracterização das transformações em que um determinado componente *Kettle* foi utilizado	DimCaseGrupo (Normal)
3	Componente	Identificação e caracterização dos componentes *Kettle* utilizados e respetivas etapas do ETL	DimComponente (Normal)
4	Processo	Informação detalhada do processo ETL, nomeadamente volume de dados e janela de oportunidade.	DimProcesso_junk (*Junk*)

A única tabela de factos que compõe o esquema dimensional, "*FT_PontoNegro"*, tem como medidas a frequência absoluta de utilização ("*freqAbs"*), o número de transformações em que aparece ("*freqCase"*), o número máximo de vezes em que um determinado componente foi utilizado numa transformação ("*caseFreq"*), a duração total ("*duracaoTotal"*), a mediana da duração ("*duracaoMediana"*), a duração média ("*duracaoMedia"*), a duração mínima ("*duracaoMin"*) e a duração máxima ("*duracaoMax"*). Todas estas medidas foram definidas de acordo com os dados que foram extraídos pela ferramenta *Disco*, conforme se percebe pela Tabela 4, que mostra a correspondência entre o nome das métricas na ferramenta de *process mining* e o nome atribuído à respetiva medida no *data warehouse*. Além destas, foram acrescentadas duas outras medidas de controlo, denominadas "*bemEstar_parcial"* e "*isBlackpoint"*, cujo significado será esclarecido aquando da apresentação da expressão final do cálculo do índice de bem-estar. De seguida, na Figura 13, apresenta-se o esquema dimensional elaborado em *Indyco* [16], tendo os vários elementos de dados do esquema representados através da notação de Golfarelli [17].

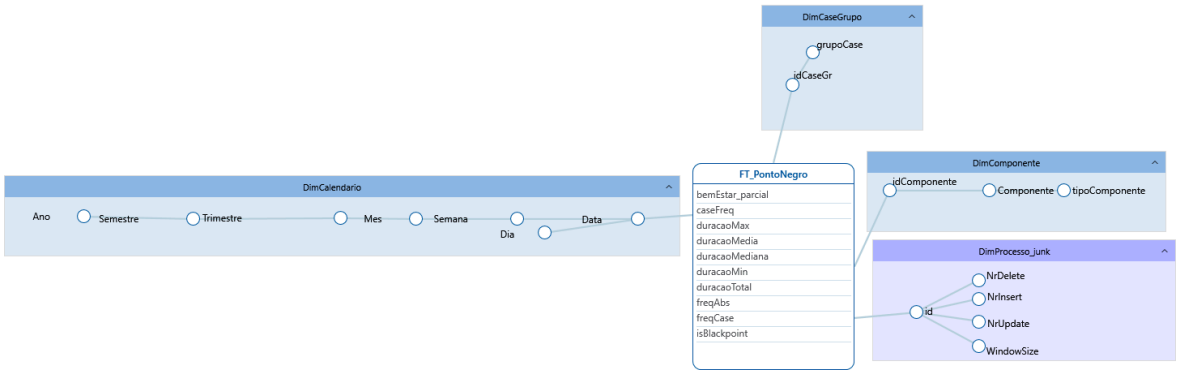


Figura 13 – Esquema dimensional "*FT_PontoNegro"*, elaborado em *Indyco*.

Tabela 4 – Correspondência entre o nome da métrica em *Disco* e a respetiva medida no *data warehouse.*

Métrica em *Disco*	**Medida no *Data Warehouse***
Absolute frequency	freqAbs
Case frequency	freqCase
Max. repetitions	caseFreq
Total duration	duracaoTotal
Median duration	duracaoMediana
Mean duration	duracaoMedia
Min. duration	duracaoMin
Max. duration	duracaoMax

3.4 Processo ETL para povoamento do *Data Warehouse*

Depois de analisado o *output* fornecido pela ferramenta de *process mining* selecionada e tendo implementado o *data warehouse* de acordo com a estrutura proposta no ponto anterior, passámos à implementação do processo ETL em causa. Para isso, tivemos em atenção as necessidades de enriquecimento de dados apontadas anteriormente, bem como as medidas e dimensões constantes do esquema dimensional implementado. Como já referido, neste trabalho optámos pela utilização de ferramentas *freeware,* sempre que possível, e, como o sistema ETL exemplo está implementado em *Pentaho Data Integration*, para o novo processo ETL decidimos seguir a mesma linha de atuação e, portanto, a mesma tecnologia.

Em termos da definição da arquitetura e do planeamento geral do sistema ETL, seguiu-se os procedimentos mais habituais neste tipo de processo, especialmente naquilo que dizia respeito à pré-análise dos dados de *input,* utilizando ferramentas de *profilling* para verificar possíveis valores inesperados, e à extração dos dados para tabelas de trabalho relacionais, conhecidas como tabelas

de auditoria. Depois, com essas tarefas realizadas, desenvolvemos e realizámos o processo de enriquecimento dos dados, sem interferir diretamente com os sistemas operacionais envolvidos no desenvolvimento da solução.

Figura 14 – Processo ETL desenvolvido para o povoamento do *data warehouse*.

O processo ETL inicia-se, então, com a tarefa de extração dos dados da fonte para a tabela de auditoria envolvida, passando depois para o povoamento das várias dimensões e para a inserção dos registos na tabela de factos. Por fim, o processo ETL termina com uma transformação de manutenção, que é responsável pela limpeza da tabela de auditoria (Figura 14).

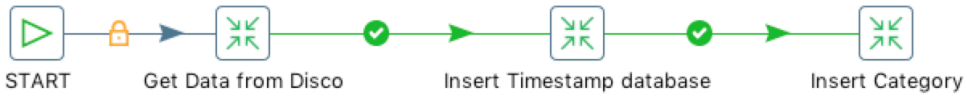


Figura 15 – Visão geral do processo de extração de dados do ficheiro *.csv* obtido de *Disco*.

Em termos do processo de extração propriamente dito, foi necessário, em primeiro lugar, extrair os dados do ficheiro *CSV* que continha os dados resultantes da análise realizada pela ferramenta de *process mining*, depois marcar os registos com o *timestamp* de execução do *package* ETL e, por fim, atualizar cada um dos componentes com a etapa do processo na qual são utilizados (Extração, Transformação, Limpeza ou Carregamento), assegurando assim as duas pré-condições anteriormente apresentadas (Figura 15).

O resultado final desse processo encontra-se parcialmente ilustrado na Figura 16.

idRedeMining	componente	tipoComponente	tTotal	tMin	tMax	tMean	tMedian	absFreq	caseFreq	maxRepeat	data_ETL	diaExecucao
1	Insert / Update	Carregamento	339	0	223	13	1	25	7	4	2017-03-28	3
2	Microsoft Excel input	Extração	1936	0	1668	276	1	7	5	3	2017-03-28	3
3	Table output	Carregamento	4297	0	3190	91	1	47	19	6	2017-03-28	3
4	Select / Rename values	Transformação	49	0	6	1	1	46	13	6	2017-03-28	3
5	Add constant values	Transformação	547	0	278	14	1	38	11	4	2017-03-28	3

Figura 16 – Extrato da tabela de auditoria "redeMining", resultante do processo de extração.

Uma vez feita a extração dos dados para a área de retenção, passámos à realização das tarefas que foram definidas para as etapas de transformação, limpeza e carregamento. Assim, procedeu-se, em primeiro lugar, ao povoamento das várias dimensões, tais como "DimComponente", "Dim*Case*", "DimCalendario" e "Dim*CaseGrupo*", terminando com o povoamento da tabela de factos. Além disso, na parte final do processo, realizamos uma pequena operação de transformação para manutenção do sistema, cuja função é simplesmente proceder à eliminação dos dados das tabelas de auditoria utilizadas ao longo do processo. Porém, antes de fazer o povoamento da tabela de factos, foi necessário implementar uma outra operação de transformação, com uma complexidade maior.

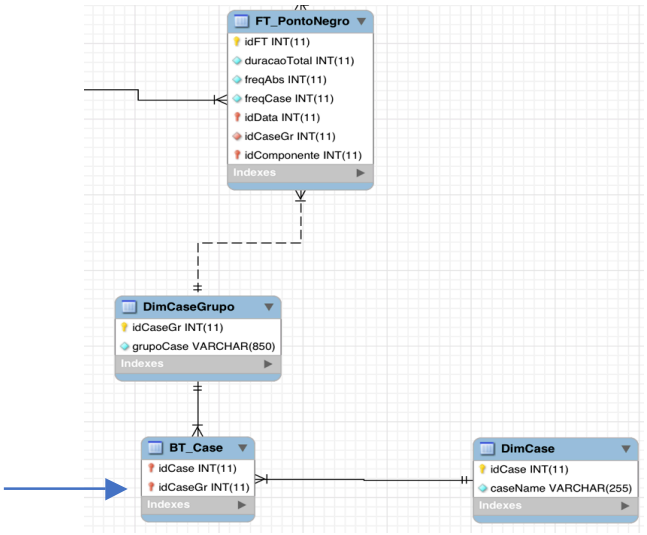


Figura 17 – Extrato do modelo lógico do *data warehouse*, com destaque para a tabela ponte, "BT_Case", implementada.

De facto, quando quisemos incluir a dimensão "Dim*Case*", ou seja, acolher a informação relativa às transformações que cada componente tinha realizado, fizemos com que fosse necessário definir um relacionamento N:N no esquema dimensional. Isto deve-se ao facto de que, quando um componente é utilizado em uma ou mais transformações e uma transformação tem um ou mais componentes associados, é necessário garantir a sua correta associação com o evento registado. Assim, tivemos que implementar uma tabela ponte para resolver esse problema (Figura 17). Contudo, este tipo de solução não é suportado pela ferramenta de *dashboarding*, o que fez com que se tivesse, também, de implementar uma outra dimensão – "Dim*CaseGrupo*" -, que, para cada ponto negro, disponibiliza a lista de transformações nas quais o componente foi utilizado. As

dimensões "Dim*CaseGrupo*" e "Dim*Case*" não levantaram qualquer problema em termos de povoamento, tendo este ocorrido de forma bastante regular através de uma implementação bastante simples. Posteriormente, foi necessário organizar o atributo de acolhimento dos dados do relacionamento, aplicando uma máscara para separar os valores do tipo *string* em vários *tokens* (pelo separador ",") e preencher a tabela ponte com as respetivas chaves estrangeiras, conforme ilustra o esquema representado na Figura 18. Este processo foi implementado através de um *stored procedure* em *MySQL*, conforme a Figura 19.

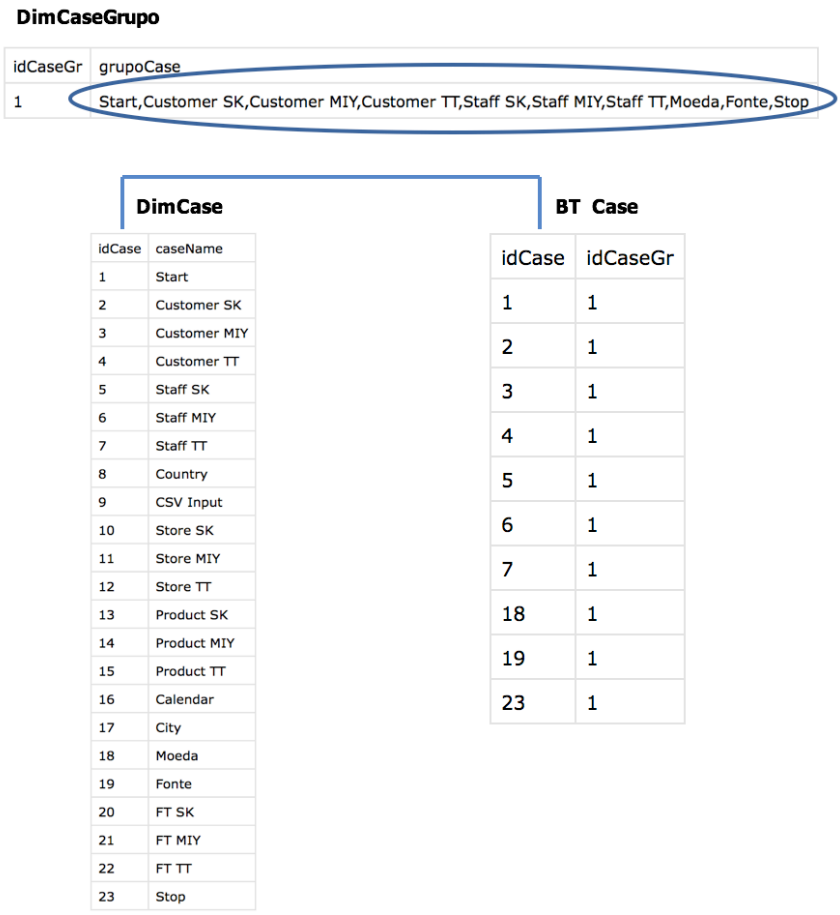


Figura 18 – Mecanismo de funcionamento da tabela ponte, "BT_Case".

```
set @size=(select count(*) from AluguerDw.redemining);
drop procedure caseGroupProcedure;
set @atual=1;
DELIMITER $$
CREATE procedure aluguerDW.caseGroupProcedure()
BEGIN
    DECLARE x INT Default 0 ;
  SET x = @atual;
  WHILE x <= @size DO
    set @caseGroup=(select GROUP_CONCAT(distinct(trace) SEPARATOR ',') from log where componenteKettle in
        (select componente from redemining where idRedeMining=@atual) and executionOrder in
        (select diaExecucao from BlackpointDW.controlo));
    set @atual=@atual+1;
    INSERT ignore INTO  BlackpointDW.DimCaseGrupo (grupoCase) VALUES(@caseGroup);
    set @caseID=(select idCaseGr from BlackpointDW.DimCaseGrupo where grupoCase=@caseGroup);
    INSERT INTO  BlackpointDW.audCaseGroup (caseGroup,idFTCaseGroup) VALUES(@caseGroup,@caseID);
    SET x = x + 1;
  END WHILE;
END $$
```

Figura 19 – *Stored procedure* para partição do atributo "trace" e povoamento da dimensão "*Case*Grupo" e da tabela de auditoria "aud*CaseGroup*".

A Figura 20 ilustra a transformação que é realizada no povoamento da tabela de factos. Nesta figura podemos ver que este processo de transformação começa por extrair os dados da tabela de auditoria e, depois, obtém as chaves estrangeiras para os registos das dimensões "DimCalendario", "DimComponente" e "Dim*CaseGroup*". Por fim, o processo de transformação insere os registos transformados na tabela de factos.

Figura 20 – Sequência de passos para povoamento da tabela de factos.

3.5 A Base de Dados Multidimensional

Um dos objetivos que foram definidos para este projeto de dissertação foi o de desenvolver uma expressão de cálculo para um índice que traduza o "bem-estar" do processo ETL ao longo do tempo. Com efeito, com isto pretende-se que, dado um conjunto de fatores resultantes do processo de mineração, seja possível, de uma forma intuitiva, identificar potenciais situações anómalas ao longo dos vários ciclos de execução de um sistema ETL. Para isso, foi necessário implementar uma estrutura multidimensional de suporte, que permitisse agilizar o processo de cálculo, de acordo com as várias dimensões existentes. Além disso, teve-se que definir, também, um *threshold* mínimo adequado, para que não fossem considerados "falsos" pontos negros. Por

isso, foi imperativo verificar o impacto global de cada ponto no processo ETL, bem como averiguar se todas métricas resultantes da etapa de *process mining* eram relevantes para o processo ou se, pelo contrário, estariam a distorcer os seus resultados (atentar aos *outliers* no caso da utilização de tempos médios).

```
<Schema name="BlackpointDW">
  <Cube name="BlackpointDWCube" visible="true" cache="true" enabled="true">
    <Table name="FT_PontoNegro">
    </Table>
    <Dimension type="TimeDimension" visible="true" foreignKey="idData" highCardinality="false" name="
    DimCalendario">
      <Hierarchy name="Time by Month" visible="true" hasAll="true" primaryKey="idData">
        <Table name="DimCalendario">
        </Table>
        <Level name="Data" visible="true" table="DimCalendario" column="Data" nameColumn="Data" type="Integer"
        uniqueMembers="false" levelType="TimeYears" hideMemberIf="Never">
        </Level>
        <Level name="Semana" visible="true" table="DimCalendario" column="Semana" nameColumn="Semana" type="
        String" uniqueMembers="false" levelType="TimeWeeks" hideMemberIf="Never">
        </Level>
```

Figura 21 – Extrato do ficheiro *.xml* para criação do esquema "*BlackpointDW*".

Relativamente à implementação da base de dados multidimensional, tendo o *Data Warehouse* sido implementado numa instância local de *MySQL Workbench*, foi necessário recorrer a tecnologia alternativa para que se conseguisse obter a base de dados multidimensional (um cubo OLAP), uma vez que o *MySQL* não tem capacidade de o implementar no seu ambiente. Assim, teve-se que recorrer à utilização de um servidor OLAP desenvolvido pela *Pentaho*: o sistema *Mondrian* [18]. Este sistema totalmente implementado em Java, interpreta *queries* em linguagem MDX [19], lendo os dados de uma base de dados relacional, e apresenta os resultados num formato multidimensional, tudo isto recorrendo a uma API de Java. Assim sendo, e ao contrário de outros SGBD como por exemplo *SQL Server*, *MySQL* não permite derivar diretamente esquemas relacionais em bases de dados multidimensionais. Por isso mesmo, foi necessário recorrer a mecanismos externos, mas que se analisarmos em detalhe são similares aos mecanismos implementados pelas outras opções tecnológicas, com a diferença que não temos uma interface direta planeada para o efeito. No entanto, o ficheiro *.xml* que precisamos de gerar permite a *Mondrian* identificar a fonte de dados a que fazemos referência e a forma como queremos implementar a estrutura multidimensional, identificando e caracterizando medidas, dimensões e respetivas hierarquias, bem como a forma como são obtidas a partir da estrutura relacional original. Sendo a semântica dos ficheiros *.xml* um pouco complexa do ponto de vista da deteção de erros, foi desenvolvida uma ferramenta que disponibiliza uma *interface* intermédia capaz de gerar o esquema *.xml* (Figura 21), sem que quem está a definir a infraestrutura tenha a necessidade de

escrever em *xml.* Essa ferramenta (Figura 22), chamada *Schema Workbench* [20], permite delinear a infraestrutura multidimensional e até importar esquemas já existentes e identificar incorreções.

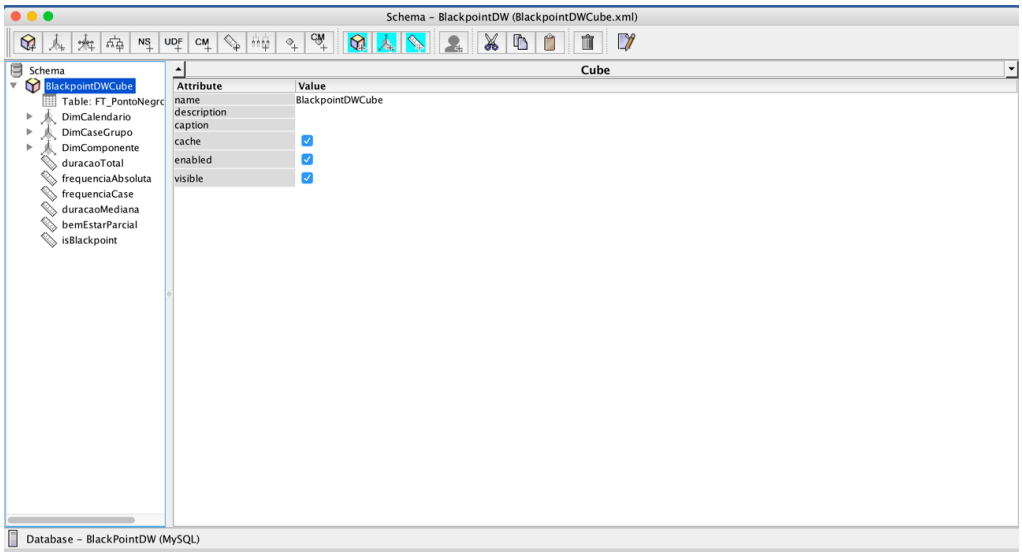


Figura 22 – *Screenshot* da ferramenta *Schema Workbench*.

Obviamente que, antes de se criar o esquema para o cubo, foi necessário definir a sua estrutura. A estrutura do cubo definido integra três dimensões e uma tabela de factos. As dimensões integradas foram "DimCalendario", "DimCaseGrupo" e "DimComponente". Todas estas dimensões foram obtidas diretamente do *data warehouse* já implementado. Relativamente às medidas utilizadas no cubo, estas foram definidas a partir do conjunto de medidas estabelecido anteriormente, em particular das métricas resultantes da descoberta do processo na ferramenta *Disco* ("*duracaoMediana"*, "*duracaoTotal"*, *"freqAbs"* e *"freqCase"*). No que diz respeito às medidas "*bemEstarParcial"* e "*isBlackpoint"*, a sua utilização será explicada posteriormente, aquando da abordagem do cálculo do índice de bem-estar. Seguidamente, na Tabela 5, apresenta-se a lista final das medidas utilizadas e respetivas funções de agregação.

Tabela 5 – Lista de medidas respeitantes ao cubo *"BlackpointDWCube"*.

Medida	**Função de agregação**
duracaoTotal	*Avg*
frequenciaAbsoluta	*Sum*
frequenciaCase	*Avg*
duracaoMediana	*Sum*
bemEstarParcial	*Sum*
isBlackpoint	*Sum*

O esquema que se segue (Figura 23) permite perceber a estrutura do cubo "BlackpointDWCube" que foi implementado. Assim sendo, terá três dimensões: "CaseGrupo", "Calendário" e "Componente", sendo que o cubo será calculado com base em seis medidas: "duracaoTotal", "frequenciaAbsoluta", "frequenciaCase", "duracaoMediana", "bemEstarParcial" e "isBlackpoint".

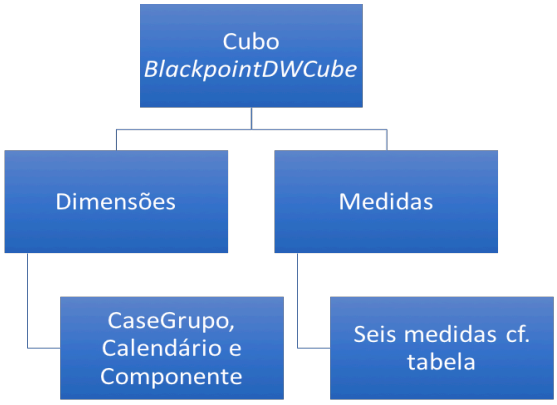


Figura 23 – Estrutura do cubo *"BlackpointDWCube"*.

No que diz respeito às hierarquias que compõem cada uma das dimensões tem-se:

DimCalendario
H1: Data -> Semana -> Trimestre -> Semestre -> Ano -> All
H2: Data -> Dia

DimCaseGrupo
H1: IDCaseGr -> CaseGroup -> All

DimComponente
H1: IDComponente -> Componente -> TipoComponente -> All

DimProcesso_junk
H1: Id -> NrInsert -> All
H2: Id -> NrUpdate -> All
H3: Id -> NrDelete -> All
H4: Id -> WindowSize -> All.

As hierarquias que compõem a dimensão temporal, "DimCalendario" (cf. Anexo I), permitem agregar ou desagregar as medidas pela data em que foram recolhidas, pela semana do ano, pelo trimestre ou pelo semestre do ano ("H1"). Por outro lado, podemos agregar a data pelo dia do mês ("H2").

A hierarquia "H1" da dimensão "DimCaseGrupo" (cf. Anexo III) permite-nos agregar as métricas de cada ponto negro pela lista de transformações *Kettle* em que foi utilizado ("CaseGroup").

No que diz respeito à hierarquia "H1" da dimensão "DimComponente" (cf. Anexo II), torna possível agregar cada ponto negro pelo componente *Kettle, "Componente",* na sua génese, bem como pela etapa do processo ETL diretamente associada ao componente ("TipoComponente").

Para concluir, na única hierarquia da dimensão "DimProcesso_*junk*" (cf. Anexo IV) agregamos as métricas de cada ponto negro pelo volume de dados, ou seja, pelo número de registos que irão ser inseridos no âmbito de um ciclo de execução ("H1"), pelo número de registos que irão ser atualizados num ciclo de execução ("H2"), pelo número de registos que irão ser "removidos" num ciclo de ETL ("H3") ou, ainda, pela proporção ocupada da janela de oportunidade ("H4"). Não esquecer que o destino distinto dos registos implica seguir caminhos diferentes no processo ETL, daí o interesse de analisarmos os dados recolhidos desta perspetiva.

Os atributos que compõem as dimensões e hierarquias do cubo são precisamente os mesmos que já integravam o *data warehouse* implementado, uma vez que serão interessantes considerar em análises futuras do processo. Se não reparemos: na dimensão calendário os atributos dia, semana, trimestre, semestre e ano são considerados elementares para uma análise do ponto de vista temporal. Por outro lado, queremos saber o impacto que um dado ponto negro tem num processo ETL, daí precisarmos de saber, entre outras coisas, a lista de transformações em que é utilizado ("*CaseGroup*" da dimensão "Dim*CaseGroup*"). Na origem de um ponto negro está sempre um componente, ou seja, um construtor elementar do processo ETL, daí requerermos o atributo "Componente" da dimensão "DimComponente". Além disso, saber a fase do processo ("TipoComponente") a que está associado esse componente pode ter bastante interesse numa análise futura, sobretudo se quisermos avaliar possíveis alternativas de implementação. Por último, saber mais alguns detalhes ligados a um determinado ciclo de execução do nosso ETL, como por exemplo o volume de dados, só contribuirá para uma análise mais detalhada e permitirá, mais do que isso, prever execuções futuras com um maior grau de certeza. Por tudo isto, sugerimos a consulta dos anexos para verificar o significado de cada um dos atributos.

Capítulo 4

Um Índice de Bem-Estar para ETL

4.1 A Definição do Índice

As variáveis que resultaram da realização do processo de mineração de dados sobre os registos de funcionamento do sistema de ETL são bastante complexas de analisar, em termos individuais, se não for possível correlacioná-las de alguma forma. Uma das maneiras possíveis de o fazer é através da definição de uma função matemática que, com base no conjunto de variáveis em questão, retorna um valor numérico, representativo do "bem-estar" do processo de ETL. Desta forma, é possível traçar uma série temporal que permita ilustrar a sua variação e, ao mesmo tempo, identificar de forma mais rápida, uma situação potencial anómala, sem que para isso seja necessário monitorizar variável a variável. A definição de um índice como este representa uma mudança clara de paradigma, uma vez que o processo de análise do bem-estar do sistema de ETL passa a ocorrer de forma inversa, isto é, verifica-se, primeiro, o valor do índice e, mais tarde, caso se justifique, faz-se uma análise complementar variável a variável.

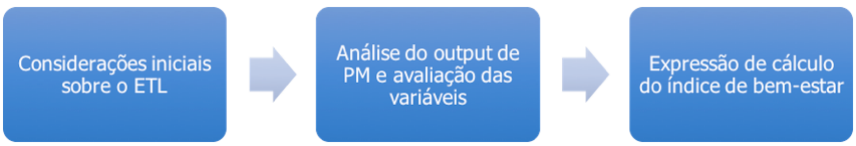


Figura 24 – Sequência de passos para desenvolvimento de uma expressão de cálculo do índice de bem-estar.

Para que seja mais fácil entender a forma como desenvolvemos a expressão matemática para o cálculo do índice de bem-estar, apresentamos, na Figura 24, um esquema com a sequência das atividades que foram desenvolvidas nesse processo de cálculo.
O desenvolvimento do índice de bem-estar emergiu da pretensão de avaliar o desempenho de um sistema ETL ao longo do tempo. Todavia, é um processo que levanta algumas questões fundamentais, que não podem ser ignoradas. De referir, o volume de dados envolvido na invocação do *package*, a janela de oportunidade disponível e a respetiva proporção de ocupação pela execução do sistema de ETL, o *threshold* e a sua influência na identificação de pontos negros, a avaliação da adequabilidade da duração média como métrica integrante da expressão final e o consumo energético do sistema.

O volume de dados envolvido no processo de invocação do *package* ETL condiciona irremediavelmente a *performance* do sistema de ETL. Com efeito, o volume de dados envolvido numa execução mede-se, de forma genérica, através do número de registos envolvidos. Neste caso, e visto que se pretende aprofundar o conhecimento acerca do sistema de ETL, os dados são analisados tendo como base o tipo de operação a que os novos dados estarão sujeitos. Assim, separaram-se os dados em três grupos distintos: registos novos (*insert*), registos para atualização (*update*) e registos marcados para remoção (*delete*). De salientar que, nem sempre se admitem operações de *insert* ou *update* sobre determinadas tabelas de dimensão. No entanto, esse tipo de decisões é tomado previamente, na fase de planeamento da arquitetura do sistema, uma vez que, usualmente, implica processos distintos de tratamento no sistema ETL global.

A janela de oportunidade é um fator preponderante em qualquer sistema ETL, uma vez que representa o intervalo de tempo que foi estabelecido para a execução de todos os processos ETL e, consequentemente, para a disponibilização dos dados mais recentes no *data warehouse* que o sistema alimenta. A janela temporal é um elemento determinante no sistema de ETL. Quanto maior for o tempo de execução total do ETL, menor será o tempo remanescente para uma eventual repetição do processo, já que a janela temporal é normalmente estática. Todavia, em termos ideais, assume-se que o tamanho da janela de oportunidade deverá ser suficiente para permitir a ocorrência de uma execução regular e, em caso de falha, de uma nova tentativa de execução. Em suma, sempre que possível, a janela de oportunidade deverá ser definida de forma a permitir duas execuções do processo.

O *threshold* define o valor mínimo a partir do qual se considera, no contexto de um dado sistema ETL, que um determinado construtor atuou como um ponto negro durante o seu ciclo de execução. Assim, a determinação do *threshold* não pode ser realizada de forma aleatória e deverá ter em conta o contexto de uma execução considerada normal de um dado processo ETL. Desta forma poder-se-á fazer uma distinção correta entre possíveis casos anómalos (ou não), devendo o seu valor resultar da experiência de utilização do sistema pelo utilizador.

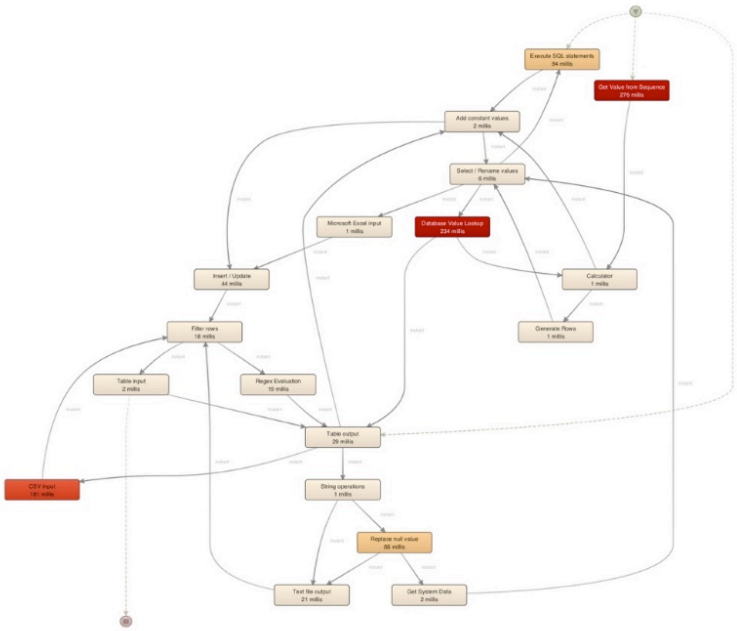


Figura 25 – Rede de execução de um ciclo de ETL, analisada na perspetiva do tempo médio.

Quando se analisam as métricas resultantes de um processo de mineração, uma questão fundamental que se levanta é a influência de *outliers* na deteção dos pontos negros. Na definição de um *outlier* incluem-se todas as medições de tempo anómalas que fazem parte do cálculo do tempo médio de processamento de cada passo do sistema ETL. Ora, ao fazerem parte da expressão de cálculo, estes poderão estar a contribuir para uma identificação errada de pontos negros, o que pode conduzir a uma hipotética sobrevalorização de um determinado ponto de execução. Assim, e tendo em conta quer o facto de *Disco* calcular sempre a média e a mediana do tempo, quer a própria natureza do processo que está a ser analisado, achamos que a utilização de uma mediana é a melhor forma para calcular o tempo que cada passo do sistema ETL demora a

ser executado, uma vez que pela própria fórmula de cálculo, exclui valores anormalmente elevados, bem como valores anormalmente baixos [21].

As Figuras 25 e 26 ilustram precisamente o comportamento que acabámos de descrever. Como facilmente se intui a utilização do tempo médio como filtro de análise, ao invés da mediana do tempo, identifica pontos negros que, na realidade, não passam de "falsos" pontos negros (representados na figura com uma cor mais escura), uma vez que são influenciados por um valor anormalmente elevado entre os vários valores recolhidos ao longo dos diversos ciclos de execução do sistema ETL. As redes apresentadas nas figuras referidas são idênticas, apenas se distinguem pelo tipo de filtro de visualização que foi utilizado.

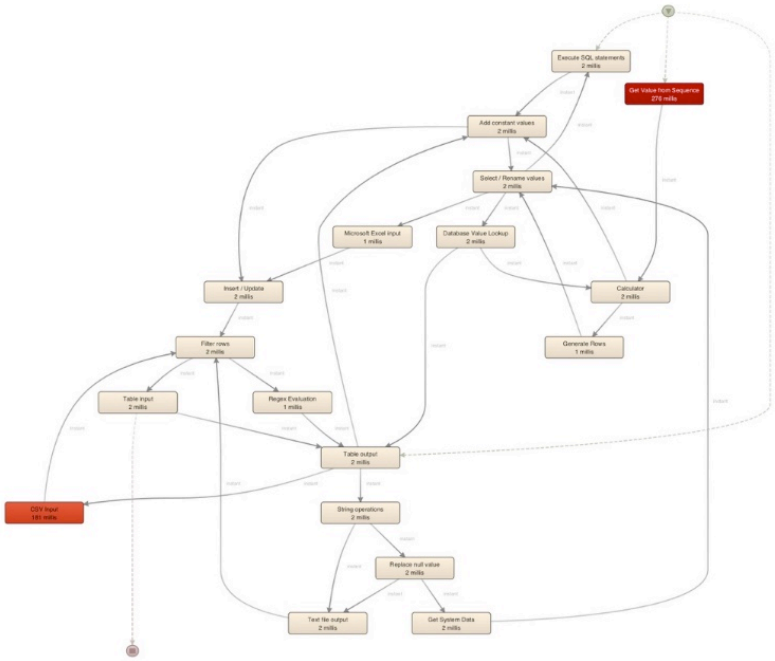


Figura 26 - Rede de execução de um ciclo de ETL, analisada na perspetiva do tempo mediano.

Além dos aspetos referidos, tomámos também em consideração o consumo energético que o sistema ETL pode fazer. Como sabemos, hoje as questões energéticas assumem particular importância, sobretudo por se considerar que os efeitos do aquecimento global são já uma evidência. Como tal, decidimos abordar também este aspeto na definição no nosso índice de bem-estar de um sistema ETL, tomando em consideração que quanto maior for a poupança de energia, maior será o "bem-estar" do sistema. Com base no trabalho apresentado em [22] definimos as várias variáveis que poderiam fazer variar o índice em termos da energia que um sistema ETL poderá consumir.

Em termos de volume de dados, teve-se também em conta que, num ciclo de execução, usualmente o número de registos novos, resultantes de operações de inserção (*insert*), supera em larga maioria os registos provenientes de operações de atualização ou de remoção. Tendo isso em consideração, definiu-se o seguinte quociente:

$$\frac{\#insert}{\#insert+\#update+\#delete}$$

Este quociente traduz o impacto dos novos registos entre o total de registos a processar pelo sistema ETL. A janela de oportunidade não tem particular relevância na expressão de cálculo, uma vez que esse tipo de situações não se traduz numa degradação direta do "bem-estar" do sistema, a não ser que deixe de ser suficiente para duas execuções do ETL. Assim, se a janela de oportunidade for suficiente para duas execuções do sistema ETL, podemos assumir o valor constante '1' para esse fator. Caso contrário, teremos que utilizar a razão entre o tempo total medido e o tempo da janela de oportunidade, medindo-se, dessa forma, a taxa de ocupação da janela. Além disso, também se considerou que cada construtor utilizado no sistema ETL contribui, em cada ciclo de execução, para o índice de bem-estar. O índice de bem-estar global é, pois, o resultado do somatório de um conjunto de contribuições respeitantes a cada um dos construtores. Assim, todas essas contribuições devem ser somadas, tendo em conta a restrição de que só serão considerados contributos de componentes cujo tempo de execução mediano tenha sido superior ou igual ao *threshold* definido.

Para calcularmos o impacto de cada componente ETL utilizado, o dito contributo basal de cada elemento que compõe a arquitetura do nosso sistema, definimos a expressão $\left(\frac{f_{abs}}{case_{freq.}}\right)$, que permite determinar o número médio de construtores de um determinado tipo, por etapa do sistema ETL. Ao se multiplicar esse quociente pela duração mediana fica-se a saber o tempo mediano ocupado por esse componente numa dada etapa do processo. Para terminar a análise da sua contribuição, cada construtor é utilizado numa fase específica do processo (extração, transformação ou carregamento), daí que a proporção de componentes de um determinado tipo numa determinada fase seja, de igual forma, um fator contributivo para este índice.

O consumo de energia é o último dos fatores a considerar na determinação do índice. Os valores de consumo utilizados são relativos aos diversos componentes utilizados na implementação do sistema ETL global. Nesse sentido, e recorrendo ao trabalho desenvolvido no âmbito da

categorização do consumo de energia em sistemas de povoamento de *data warehouses*, é possível avaliar o nível de consumo de cada componente ao longo do tempo nos vários ciclos de execução. No entanto, e dado o facto de se tratar de um trabalho desenvolvido por diferentes autores, optou-se por estimar o consumo, em *joules*, para cada um dos construtores utilizados, uma vez que o cálculo automático do consumo de energia não faz parte do trabalho desta dissertação. Como referido anteriormente, este cálculo poderia ser realizado tal como se apresenta em [22].

De seguida, apresenta-se a expressão final, (1), para a determinação do índice de bem-estar de um sistema ETL, bem como três outras sub expressões auxiliares de cálculo, mais precisamente, (2), (3) e (4). A expressão (2) representa precisamente a condição anteriormente apresentada para a janela de oportunidade, ou seja, se a janela de oportunidade for suficiente para duas execuções do sistema ETL, podemos assumir o valor constante '1' para esse fator. Caso contrário, teremos que utilizar a razão entre o tempo total medido e o tempo da janela de oportunidade, medindo-se, dessa forma, a taxa de ocupação da janela. A expressão (3) avalia o impacto dos componentes utilizados numa determinada etapa do processo ETL, face à totalidade de componentes utilizados. Por último, a expressão (4), representa o cálculo do consumo de energia para cada um dos componentes distintos utilizado no processo.

$$índice_{bem-estar} = \frac{\#insert}{\#insert + \#update + \#delete} \times f\left(\frac{t_{total} \times 2}{janela_{oport.}}\right) \times \sum_{dur_{mediana} \geq threshold} p.\left(\frac{f_{abs}}{case_{freq.}}\right).dur_{mediana} + E \quad (1)$$

$$f(x) = \begin{cases} 1 \; se \; x \leq 1 \\ \left(\frac{t_{total}}{janela_{oport.}}\right) \; se \; x > 1 \end{cases} \quad (2)$$

$$p_i = \frac{\#componentes \; etapa \; i}{\#componentes \; total}, \quad (3)$$

$$i \in \{extração, transformação, carregamento, limpeza\}$$

$$\mathrm{E} = \mathrm{P}.\Delta\mathrm{t}, \quad (4)$$

Por observação da expressão (1), acima apresentada, podemos verificar que o resultado do cálculo é um número real positivo. No entanto, tal resultado complica o processo de análise e comparação. Além disso, também podemos verificar que, tendo em conta a reduzida variação de fatores como a frequência absoluta de utilização de um componente, a frequência de *case* e, consequentemente, do peso associado ao conjunto de componentes numa determinada etapa, o índice tenderá a aumentar com o aumento da duração mediana e da energia consumida durante a execução do sistema ETL. Ou seja, o aumento do índice de bem-estar pressupõe uma degradação em termos de performance do *package* global do sistema ETL. No entanto, quer para possibilitar uma análise comparativa, quer para facilitar uma possível representação gráfica dos seus valores, fizemos a normalização do resultado da expressão de forma a que ficasse sempre contido no intervalo [0,1]. Para que isso fosse possível, utilizámos a seguinte função polinomial (5):

$$g(x) = \frac{x}{x^2 + 1} \qquad (5)$$

Deve-se sublinhar que, à medida que o valor de x aumenta, o valor de $g(x)$ tende para valores próximos de 0. Em termos matemáticos, tal tendência é representada pelo conceito de limite – ver expressão seguinte. De notar que no caso do índice de bem-estar, os valores serão sempre positivos.

$$\lim_{x \to +\infty} g(x) = \lim_{x \to +\infty} \left(\frac{x}{x^2 + 1} \right) = 0$$

Assim, quando se representar graficamente a variação do índice de bem-estar normalizado, $g(x)$, a análise terá de ser contrária à que apresentámos anteriormente, isto é, os valores mais próximos de '0' de $g(x)$ corresponderão a situações de degradação de desempenho do sistema, enquanto que os valores mais próximos de '1' corresponderão a situações de execução regular do sistema ETL. Daí a designação de "bem-estar".

Vejamos agora a questão de uma situação de bem-estar parcial. O cálculo parcial do índice de bem-estar corresponde ao cálculo da parcela $p.\left(\frac{f_{abs}}{case_{freq.}}\right).dur_{mediana}$ para cada um dos construtores utilizados num dado ciclo de execução do sistema ETL. A vantagem do cálculo prévio deste passo torna-se evidente pela facilidade com que, posteriormente, ao carregar os dados para

o *data warehouse,* e mais tarde para uma estrutura multidimensional de dados, apenas temos que utilizar a função de agregação de soma (SUM()) e retirar, de forma automática, o valor do índice para um dado ciclo de execução do sistema ETL. Desta forma, consegue-se minimizar a complexidade das interrogações que serão necessárias para a manipulação das estruturas multidimensionais de dados e apresentação da variação do índice num sistema apropriado de visualização de dados – um *dashboard*.

Quando se fez a apresentação da expressão final para o cálculo do índice de bem-estar, impôs-se a restrição de que só seriam somados os contributos de cada construtor cujo valor de duração mediana fosse superior ou igual ao *threshold* definido. Porém, para que depois se pudesse efetuar o cálculo final, introduziu-se também uma nova medida – *isBlackpoint* -, agregável pela função de agregação *SUM()*. Esta medida, que assume apenas valores discretos, mais precisamente 0 ou 1, permite verificar se o construtor foi identificado (1) ou não (0) como um ponto negro, ao mesmo tempo que possibilita contabilizar o número de pontos negros detetados num determinado intervalo temporal, por exemplo.

4.2 Previsão do Valor do Índice

Na altura em que se propôs a criação de um índice de bem-estar como mecanismo de monitorização para um sistema ETL e, por conseguinte, como mecanismo primário para a verificação de anomalias do seu funcionamento, não se referiu a possibilidade de analisar uma possível maneira para fazer a previsão do comportamento dos vários indicadores que suportam a criação e a manutenção do índice de bem-estar. Porém, agora, parece-nos natural fazer essa análise e introduzir a temática da previsão no contexto aplicacional do índice de bem-estar de um sistema de ETL, uma vez que a antecipação de hipotéticas situações anómalas permitirá definir e implementar alguns mecanismos de contingência, que poderão atuar sobre o sistema antes que o um dado problema possa ocorrer e causar algumas dificuldades para quem interage com o sistema. Apesar disso, até ao momento, apenas se propôs o cálculo do índice de bem-estar com base nos dados angariados durante um ciclo de execução do sistema que já tenha ocorrido. A realidade é que este novo sistema de monitorização, com funcionalidades de previsão integradas, ganharia valor se permitisse, de antemão, prever, com alguma precisão, quer o índice de bem-estar, quer os pontos negros de uma futura execução do sistema de ETL. Para que isso fosse

possível, decidiu-se recorrer a um conjunto de técnicas adequadas de *data mining*, que seriam aplicadas sobre os dados angariados em ciclos de execução anteriores. Desta maneira, conseguimos prover o sistema de índices com uma componente de previsão que atuará a partir dos dados de experiências de execução anteriores. Para fazermos a implementação deste componente decidimos adotar a metodologia *CRISP-DM* [23], uma metodologia amplamente reconhecida e utilizada na área de *data mining*.

Figura 27 – Etapas do processo de implementação do sistema de previsão.

O processo de implementação do sistema de previsão envolveu várias etapas, ilustradas na Figura 27. Assim, na primeira etapa de identificação das variáveis objetivo, foi necessário analisar cada um dos atributos que pudesse influenciar o cálculo do índice e, posteriormente, definir o tipo de modelo que se pretenda construir. Anteriormente, referiu-se a existência de vários fatores que têm influência direta no bem-estar de um sistema ETL, nomeadamente: o volume de dados envolvido na invocação do *package*, a janela de oportunidade estabelecida e respetiva percentagem de ocupação, o *threshold* definido e a sua influência na identificação de pontos negros, a avaliação da adequabilidade da duração média como métrica integrante da expressão final e o consumo energético do sistema. Destes fatores, podem-se excluir imediatamente a janela de oportunidade e o *threshold*, visto que são parâmetros que permanecem constantes ao longo da execução do sistema ETL, pelo que se tornaria irrelevante incluí-los no modelo de previsão.

Quanto à medição do tempo, a discussão sobre a duração média e a sua adequabilidade relativamente à duração mediana não tem relevância para este modelo, visto que aquilo que se pretende é fazer a previsão de um índice de bem-estar para o *package* ETL, o que faz com que a realização de um processo de análise, atividade a atividade, não faça qualquer sentido. Assim, ao invés da utilização dessas métricas, optou-se apenas pela utilização do tempo total, desde o início da execução do sistema ETL até à sua conclusão. Por fim, temos o *timestamp* associado à invocação do *package* ETL. Este é outro dos atributos que foram incluídos no modelo de previsão. Além deste atributo, o modelo de previsão considera, também, o volume de dados (separado por tipo de operação), o tempo total e o valor do índice.

O modelo de previsão idealizado é um modelo típico de regressão, uma vez que a variável objetivo (aquela cujo valor queremos prever) é do tipo numérico, suportando o valor que o índice pode assumir. Em termos de implementação propriamente dita, foram atribuídos aos atributos índice e *timestamp* os papéis de *label* (variável objetivo) e *id*, respetivamente. A inclusão do *timestamp* no modelo de previsão e a decisão sobre qual deveria ser o seu papel foi alvo de uma análise cuidada. Com efeito, sendo o sistema ETL do *cluster* de empresas alvo da área comercial, poder-se-á pensar no interesse que teria esse registo temporal, no sentido de se poder traçar tendências de execução do sistema ETL ao longo do tempo, uma vez que estudos de mercado tendem a demonstrar que, ao longo do tempo, se verificam regularmente algumas tendências cíclicas, associadas geralmente a dias de semana específicos, períodos do dia, etc. Apesar disso, o volume de dados previsto para um determinado dia e hora, já se rege, ainda que indiretamente, por essas mesmas tendências. Assim, e após a realização da fase de testes do modelo de previsão, considerando, ou não, o *timestamp* como atributo parte integrante do modelo, decidiu-se pela sua não inclusão no modelo, tendo sido apenas utilizado como *role id*, ou seja, como identificador único de um registo. Porém, caso se optasse pela sua inclusão, as previsões, mesmo como exemplos bem conhecidos, sofreriam claros desvios face aos valores reais. Os casos de treino e de teste do nosso modelo de previsão seguiram o formato apresentado nas Figuras 28 e 29.

day	insert	update	delete	t_total	index
03/04/16 22:53	17715	0	0	20707	0.01436532
12/04/16 09:17	300	0	0	6262	0.03066194
20/04/16 07:52	3000	5	1	10320	0.05255254
01/05/16 13:14	4100	3	0	18096	0.02998445

Figura 28 – Pequeno excerto dos casos de treino para o modelo de previsão do índice de bem-estar.

day	insert	update	delete	t_total	index
10/03/17 20:06	17715	0	0	20707	

Figura 29 – Pequeno excerto de um caso de teste para o modelo de previsão do índice de bem-estar.

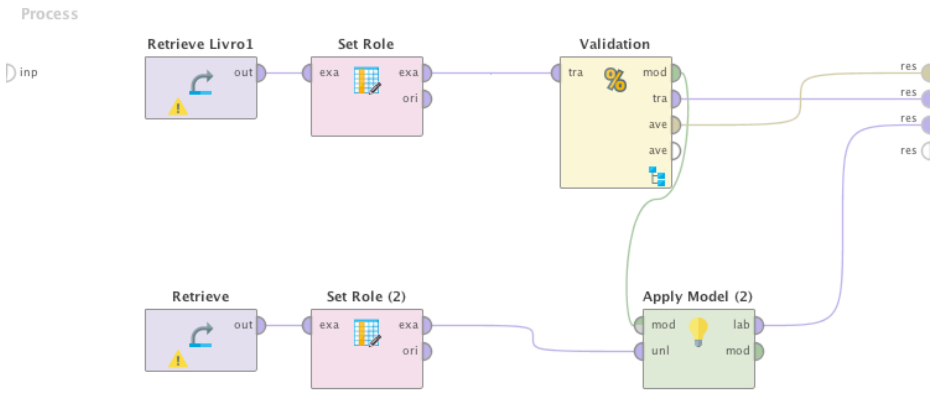


Figura 30 – Modelo previsão do índice de bem-estar: processo genérico.

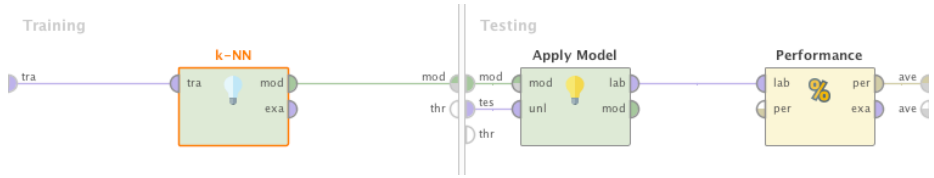


Figura 31 – Modelo previsão do índice de bem-estar: etapas de treino e teste.

Em termos de implementação do modelo, os dados de treino são obtidos através do operador *Retrieve* (Figura 30), tendo-se definido o atributo *index* como variável objetivo. Posteriormente, é realizada a validação do modelo, recorrendo-se ao algoritmo *K-Nearest Neighbour* (Figura 31) que utilizou como base de trabalho um conjunto de dados de teste. De salientar que os *screenshots* apresentados nas Figuras 30 e 31 dizem respeito à implementação do modelo para a previsão da ocorrência de pontos negros em *RapidMiner Studio* [24].

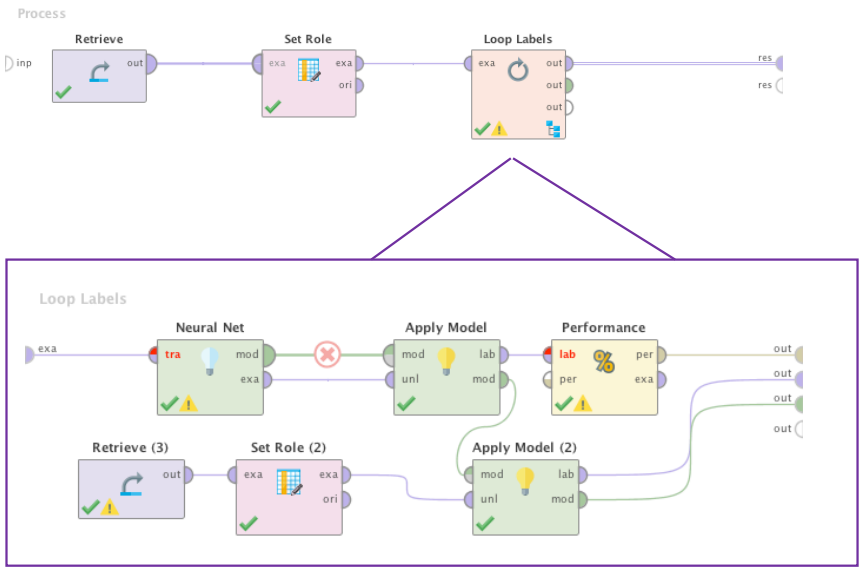


Figura 32 – Modelo previsão de pontos negros.

Para construir o modelo para a previsão de pontos negros (Figura 32) foi importante, sobretudo, definir o nível de detalhe dos casos de treino e de teste. Como tal, o instante temporal de execução do sistema ETL e o volume de dados, por tipo de operação, foram dois dos atributos imediatamente definidos para integrar o modelo. Além destes dois atributos acrescentaram-se mais dezoito, cada um deles relativo a um construtor que foi utilizado na implementação do sistema ETL. Relativamente a estes construtores, tínhamos algumas dúvidas como cumprir o objetivo definido para o modelo que queríamos estabelecer e a traduzir esse objetivo no modelo. Isso foi ultrapassado, na altura, quando decidimos utilizar atributos binominais (0 ou 1), ou seja, consoante o construtor que fosse utilizado fosse identificado como um ponto negro (1) ou não (0), num determinado ciclo de execução. Porém, mais tarde, essa solução foi abandonada porque o modelo para ser treinado teria de ser capaz de prever ambas as possibilidades, qualquer que fosse o construtor *Kettle* utilizado.

O problema é que alguns componentes nunca chegam a atuar como pontos de estrangulamento, pelo que os casos de treino a utilizar teriam de contemplar, erradamente, situações fictícias. Por isso excluímos a forma como abordámos inicialmente a resolução do problema, optando-se, então, por prever o tempo mediano de execução para cada construtor. Assim, no final do processo de previsão, basta comparar esses tempos com o *threshold* previamente definido e identificar os

pontos negros, caso se verifiquem. Neste modelo as variáveis objetivo ficaram, assim, relacionadas diretamente com cada um dos construtores do *Kettle*.

day	insert	update	delete	Execute SQL statements	Select / Rename value	Filter rows	Table output	Add constant values	Database Value Lookup	Insert / Update	Regex Evaluation
3/4/16 22:53	17715	0	0	2	2	2	2	2	2	2	2
12/4/16 9:17	300	0	0	2	2	2	2	2	2	2	2
20/4/16 7:52	3000	5	1	2	1	1	2	1	1	1	1
1/5/16 13:14	4100	3	0	2	1	1	1	1	1	1	2
15/5/16 9:30	5113	4	1	2	2	2	2	2	2	2	2
29/5/16 18:45	330	0	0	1	0	1	1	1	1	1	1
23/6/16 13:56	3600	0	0	2	2	2	2	2	2	2	2
3/7/16 17:53	3511	3	3	2	2	2	2	2	2	2	2
8/7/16 21:20	4868	2	1	1	1	1	1	1	1	1	1

Figura 33 – Pequeno excerto dos casos de treino para o modelo de previsão de pontos negros.

Em termos genéricos o modelo, apresentado na Figura 32, rege um processo de previsão bastante regular, recebendo os dados de treino como *input* (*Retrieve*), identificando depois os atributos com um papel correspondente (*Set Role*) e terminando com a previsão do valor *label* a *label* (*Loop Labels*), com recurso à utilização de uma rede neuronal (*Neural Net*). Os casos de treino apresentados na Figura 33 seguiram a linha de raciocínio anterior, ou seja, além do *timestamp* associado à ocorrência do ponto negro, e do volume de dados, representado pelo número de operações de *insert*, *update* e *delete*, acrescentam, ainda, cada componente e a respetiva duração mediana.

4.3 O Índice Previsto

A grande mais valia do processo de previsão do índice de bem-estar de um sistema ETL, como mecanismo preventivo de anomalias futuras, foi já, anteriormente, apresentada. Nesse sentido, fazer a previsão do índice de bem-estar para um dado futuro ciclo de execução, com a sua analise em termos individuais, representa uma perspetiva um pouco redutora. Assim, no sentido de avaliar a precisão das previsões efetuadas ao longo do tempo, achamos necessário fazer a implementação de um *data warehouse específico,* acompanhado pela respetiva base de dados multidimensional, para acolher os registos da evolução temporal das previsões realizadas, bem como colocar as os valores das previsões à disposição dos agentes de decisão do sistema ETL: os seus administradores. Assim, de forma semelhante àquela que já tinha sido adotada anteriormente, esboçou-se um esquema dimensional em *Indyco Builder*, após ter-se feito o levantamento e a análise dos requisitos para o *Data Warehouse*, que incluíram, nomeadamente:

- Saber qual o índice de bem-estar de um determinado sistema ETL, para um determinado período de tempo.
- Saber qual o volume de dados (*insert*, *update* ou *delete*) previsto para um determinado sistema ETL, para um determinado período de tempo.

Em termos de dimensões relevantes, e com base nos requisitos expostos, estabelecemos as dimensões "Calendário" e "Processo". A dimensão "Calendário" permitirá analisar os factos segundo uma dada perspetiva temporal, enquanto que a dimensão "Processo" permitirá identificar a que sistema ETL diz respeito o índice de bem-estar previsto. Relativamente à definição do grão da tabela de factos, ou seja, a definição do nível de informação mais básico representado no esquema, esta foi realizada de forma a poder acolher a informação relativa ao valor de um índice de bem-estar, previsto para um determinado sistema ETL, num determinado período de tempo. As medidas a incluir no *data warehouse,* quatro no total, derivaram diretamente de tudo o que já foi explicado anteriormente, e tiveram em conta na sua definição a importância do volume de dados associado ao tipo de operação, isto é, se a operação realizada foi um *insert* (*predictInsert*), *update* (*predictUpdate*) ou *delete* (*predictDelete*), durante a previsão do índice propriamente dito (*predictIndex*). Desta forma, as três primeiras medidas representam a previsão para o volume de dados da próxima execução do sistema. De forma complementar, às quatro medidas anteriores, acrescentamos uma outra medida para acolher o tempo total previsto para um determinado ciclo de execução do sistema ETL. Seguidamente, na Figura 34, apresenta-se o esquema dimensional final.

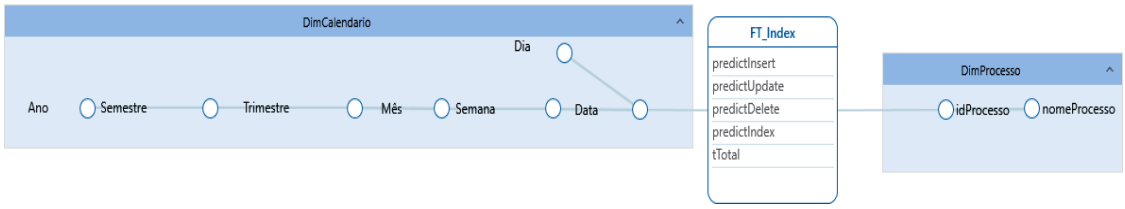


Figura 34 – Esquema dimensional para acolher as previsões do índice de bem-estar do ETL.

Para que os agentes de decisão tivessem acesso aos serviços do sistema implementado, reconhecendo as suas vantagens, projetou-se uma base de dados multidimensional, - um cubo OLAP. Em termos de estrutura do cubo propriamente dita, este tem duas dimensões e uma tabela de factos. As dimensões que farão parte da estrutura multidimensional são nomeadamente, a dimensão "DimCalendário" e a dimensão "DimProcesso" (cf. Anexo V). Neste conjunto de dimensões não há, naturalmente, grandes surpresas uma vez que estas dimensões foram derivadas diretamente do *data warehouse* anteriormente implementado. Quanto às medidas utilizadas, estas derivaram, também, do mesmo *data warehouse*, tendo sido, neste caso, apenas renomeadas. Assim, *predictInsert*, *predictUpdate*, *predictDelete* e *predictIndex* passaram a chamar-se *nrInsert*, *nrUpdate*, *nrDelete* e *predictedIndex*. O tempo total (*tTotal*) permaneceu inalterado.

Seguidamente, na Tabela 6, apresenta-se a lista final de medidas e respetivas funções de agregação.

Tabela 6 –Medidas do esquema dimensional e respetiva função de agregação.

Medida	**Função de agregação**
nrInsert	*Avg*
nrUpdate	*Avg*
nrDelete	*Avg*
predictedIndex	*Avg*
tTotal	*Avg*

No que concerne à implementação propriamente dita desta estrutura, os condicionalismos adjacentes à escolha de *MySQL Workbench* como motor de base dados para o *data warehouse*, aplicam-se, de igual forma, a este processo. Como tal, foi necessário recorrer ao *Mondrian* e à ferramenta de administração de esquemas multidimensionais da *Pentaho* para gerar o correspondente esquema *.xml*. Este esquema seria mais tarde reconhecido e acolhido pelo *Mondrian*, para suportar a estrutura do cubo referido e dos seus dados. Na definição das dimensões integradas no esquema definimos algumas dimensões para permitir a exploração dos dados entre diferentes níveis de análise e de agregação, nomeadamente, para as dimensões:

- **DimCalendario**
  H1: Data -> Semana -> Trimestre -> Semestre -> Ano -> All
  H2: Data -> Dia
- **DimProcesso**
  H1: idProcesso -> nomeProcesso -> All

A organização das hierarquias apresentadas, bem como os nodos relativos a cada um dos seus níveis de agregação são, por si, bastante óbvios, não carecendo de qualquer outra explicação adicional para além da apresentação da sua representação hierárquica. Os atributos que compõem as dimensões e hierarquias deste cubo são, naturalmente, os mesmos que integraram as dimensões do *data warehouse* anteriormente definido.

4.4 A Previsão de Pontos Negros

Um pouco à semelhança do que sucedeu com a previsão do índice de bem-estar, decidimos, também, implementar um *data warehouse* e uma estrutura multidimensional para suportar um

novo grupo de processos de análise relacionados com a evolução temporal dos pontos negros que foram previstos para uma execução do *package* ETL. Na sequência do processo de levantamento de requisitos para este novo d*ata warehouse*, determinámos que precisaríamos de ter informação de suporte para:

1. Saber quais os componentes do sistema que estão previstos como potenciais pontos negros, para um determinado período de tempo.
2. Conhecer a distribuição temporal prevista para o *package* ETL, por tipo de componente, para um determinado período de tempo.

Em termos de dimensões relevantes, que podemos extrair diretamente dos requisitos estabelecidos, definimos as dimensões "DimCalendário" e "DimComponente". A primeira destas duas dimensões permite analisar os factos de acordo com uma dada perspetiva temporal. Já a dimensão "DimComponente" permite identificar os diversos construtores *Kettle* utilizados na implementação do sistema ETL, bem como a etapa do processo em que são utilizados. Quanto ao grão do esquema, este foi definido com sendo a duração mediana prevista na execução de um determinado componente *Kettle*, para um dado sistema ETL, que será executado num determinado período temporal. A única medida que foi incluída no esquema multidimensional foi o tempo mediano da execução prevista para um determinado construtor. Através da Figura 35, podemos verificar que o esquema dimensional do *data warehouse* tem duas dimensões e uma tabela de factos.

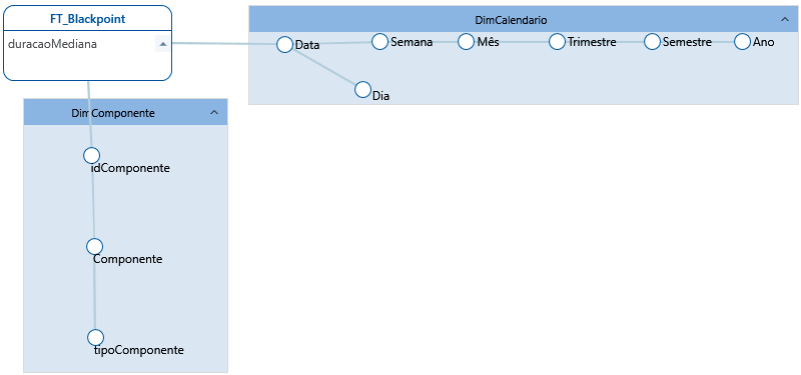


Figura 35 – Esquema dimensional do *data warehouse* de previsão de pontos negros.

Para que fosse possível disponibilizar a informação de suporte à previsão da ocorrência de pontos negros aos agentes de decisão, idealizou-se uma outra base de dados multidimensional, agora especialmente orientada para esta nova previsão. O cubo estruturado tem duas dimensões e uma

tabela de factos, acompanhando de perto a estrutura do *data warehouse* definido por último. Assim, as dimensões integradas neste cubo são, nomeadamente as dimensões "DimCalendário" e a "DimComponente". Nesta estrutura apenas incluímos uma medida que representa a duração mediana prevista para um determinado componente ("duracaoMediana"). A função de agregação escolhida foi a média (AVG()). O processo de implementação deste cubo assemelha-se muito à realizada para os dois processos anteriores, já descritos. Tal como ocorreu anteriormente, tivemos, também, que utilizar uma ferramenta auxiliar para gerar um esquema *.xml* que fosse interpretável pelo *Mondrian*, o sistema que escolhemos para o acolhimento das nossas estruturas multidimensionais de dados. As duas dimensões referidas integram na sua estrutura as seguintes hierarquias:

DimCalendario

H1: Data -> Semana -> Trimestre -> Semestre -> Ano -> All

H2: Data -> Dia

DimComponente

H1: idComponente -> nomeComponente -> tipoComponente -> All

Os atributos que integram a estrutura base das dimensões e das hierarquias apresentadas são os mesmos que integram as dimensões do *data warehouse* estabelecido.

4.5 Avaliação do Índice

Nesta altura, retomamos um dos objetivos iniciais, ou seja, o de disponibilizar os indicadores chave para a monitorização do sistema numa ferramenta de *dashboarding*. Neste momento é importante saber qual é a relevância deste passo na promoção da utilidade destes indicadores. Ora, ao disponibilizar-se o índice de bem-estar, bem como as métricas associadas de previsão num *dashboard*, possibilita-se uma vista privilegiada sobre os mais recentes dados de monitorização do sistema ETL. Nesse sentido, se necessário, tem-se as bases para se poder acionar um plano de contingência, praticamente de imediato, ao mesmo tempo que se obtém facilmente informação mais detalhada sobre a origem do hipotético problema de estrangulamento que possa ter ocorrido no processo. Essa tarefa está agora facilitada pela existência de uma estrutura multidimensional de suporte ao painel de controlo, em particular. Porém, nesta altura, não convém descurar o efeito

preventivo associado à camada de previsão, que também está incluída no *dashboard* de demonstração.

Figura 36 – Conjunto de tarefas envolvidas na implementação e disponibilização do *dashboard* final.

Para se chegar à implementação e apresentação do *dashboard* final teve-se que realizar um conjunto de tarefas bem específicas (Figura 36). Estas tarefas são bastante regulares neste tipo de processos. De qualquer forma, de seguida, expomos o trabalho que cada uma delas envolveu:

1. Levantamento de requisitos para o DB – nesta fase foi preciso descobrir quais seriam as informações, ou seja, os requisitos mais relevantes aos utilizadores finais desta plataforma. Nesse sentido, esta etapa implica a interação com os administradores dos sistemas de ETL.
2. Disponibilização dos cubos na ferramenta de *dashboarding* – nesta etapa aplicou-se o método já definido no decorrer deste projeto de dissertação, uma vez que para obter a base de dados multidimensional foi necessário criar o esquema *xml* para posterior interpretação por *Mondrian* e consequente disponibilização na ferramenta de implementação de painéis de controlo.
3. Seleção de componentes e implementação de *queries* – neste ponto, foi necessário perceber que métricas queríamos representar no *dashboard* final, de acordo com os requisitos levantados na etapa 1 e, mais do que isso, determinar qual o tipo de componente que melhor se adequava à sua representação (ex: *gauge*, gráfico de barras, etc).
4. Implementação da DB final – terminada a planificação do *dashboard*, na quarta e última fase, implementamos o produto final, sempre tendo em conta tudo o que foi estabelecido em passos anteriores.

Para fazer a implementação do *dashboard* do sistema utilizámos a ferramenta *Saiku Analytics* [25] uma ferramenta *freeware*, que no nosso ponto de vista, é bastante apropriada para acolher o caso de estudo em questão.

Na generalidade do trabalho desenvolvido até ao momento, construíram-se três esquemas *.xml* relativos a três bases de dados multidimensionais: uma representativa dos dados resultantes das execuções do ETL ao longo do tempo e as restantes duas respeitantes à previsão de pontos negros

e do índice de bem-estar, respetivamente. Para que pudéssemos implementar e, consequentemente, explorar estes sistemas de dados, apenas tivemos que tornar estes esquemas visíveis à ferramenta *Saiku*. Para isso, e depois de pôr em execução o ambiente de instalação, mais precisamente o servidor de *Saiku,* ligado através da porta 8080 de *localhost*, definimos as fontes de dados em *Saiku Analytics* e, de seguida, fizemos o *upload* dos vários esquemas para o seu ambiente de trabalho. Dada a correção dos esquemas desenvolvidos, garantida pela ferramenta intermediária que utilizámos para fazer a sua geração, estes ficaram automaticamente disponíveis na listagem de cubos do sistema e aptos a serem explorados.

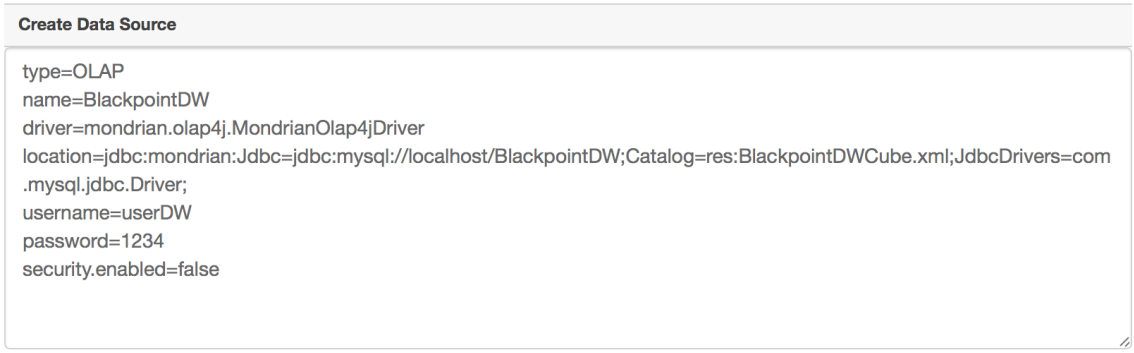


Figura 37 – Ligação de *Saiku* ao *data warehouse* "*BlackpointDW*".

Atente-se, agora, no *screenshot* apresentado na Figura 37, que foi retirado da vista para utilizadores avançados de *Saiku*. Através dele é possível ver como foi realizada a conexão com o *data warehouse* responsável pelo armazenamento dos vários pontos negros detetados ao longo dos vários ciclos de execução. Assim, depois de definido o tipo de esquema implementado e o seu nome, bastou apenas indicar a *string* com a localização do driver do sistema *Mondrian*, a localização do esquema .xml e dos drivers de *MySQL* (não esquecer que os três *data warehouses* são suportados por uma instância *MySQL*) para começar a utilizar o sistema implementado. Quanto ao *upload* do esquema .xml, o mesmo pôde ser efetuado pelo formulário disponibilizado. Também o poderíamos ter feito colocando o ficheiro do esquema nas diretorias de instalação de *Saiku,* numa localização pré-definida [26].

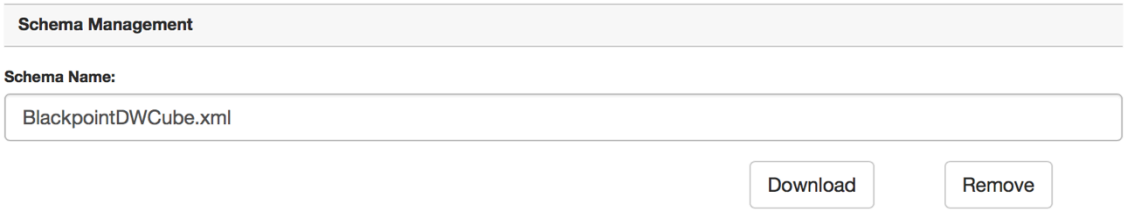


Figura 38 – *Upload* do ficheiro *xml* destinado à criação da base de dados multidimensional do *data warehouse* "*BlackpointDW*".

Na Figura 38 é possível ver que, quando o *upload* dos ficheiros é corretamente efetuado e a conexão com o *Data Warehouse* é estabelecida, os cubos são automaticamente detetados pela ferramenta de *dashboarding*. Caso as alterações ao sistema não estejam a ser reconhecidas pelo sistema, é usual reiniciar o servidor ou refrescar a cache de *Mondrian* para que isso aconteça.

Nesta altura temos, assim, disponíveis três cubos distintos e prontos a serem interrogados, nomeadamente: "*BlackpointDWCube*", o cubo respeitante ao *data warehouse* que armazena os vários pontos negros detetados ao longo dos vários ciclos de execução; "*PredictBlackPointCube*", o cubo cuja estrutura resulta do *data warehouse* implementado para a camada de previsão de pontos negros e, por último, "*PredictIndexCube*", mais precisamente, o cubo associado à camada de previsão do índice de bem-estar do sistema ETL. Esta constatação é visível na Figura 39.

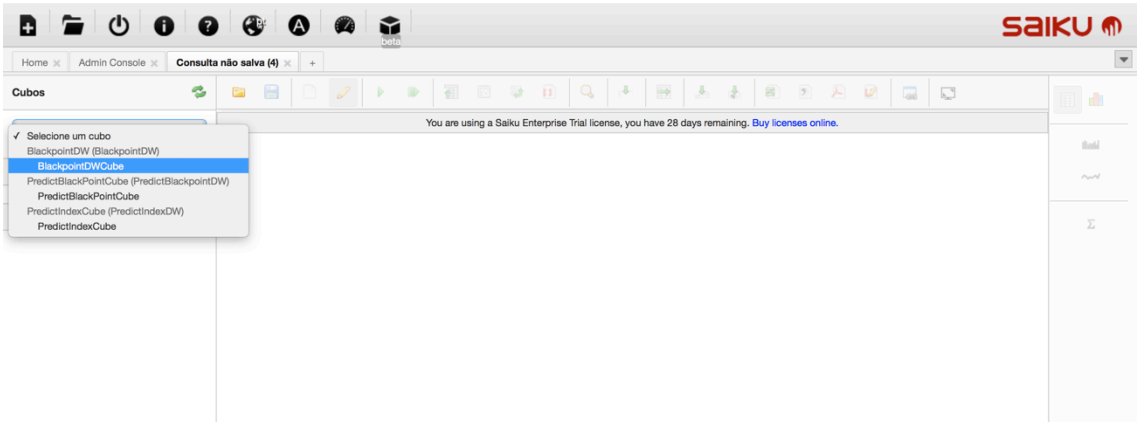


Figura 39 – Listagem de cubos disponíveis para interrogação.

Uma vez chegados a esta etapa do processo, convém relembrar os requisitos definidos anteriormente para cada um dos cubos definidos, uma vez que permitiram guiar o *design* e o conteúdo final do *dashboard*. Para isso, tivemos em consideração os seguintes requisitos:

1. Para um determinado período de tempo, saber qual o índice de bem-estar associado a uma execução do sistema ETL.
2. Para um determinado período de tempo, saber qual o número de pontos negros detetados numa execução do sistema ETL.
3. Para um determinado período de tempo, saber quais os componentes mais custosos, em termos de tempo de execução no contexto de um ciclo de ETL.
4. Prever o índice de bem-estar para um ciclo de execução do ETL a ocorrer num determinado período de tempo.
5. Prever o conjunto de pontos negros para um ciclo de execução a ocorrer num determinado período de tempo.

Para satisfazer cada um dos requisitos no *dashboard* futuro, foi necessário definir a *query* que será responsável pelo seu refrescamento e decidir de que forma seriam apresentados os resultados, de acordo com o tipo de componente mais adequado à interação com os diversos agentes de decisão. Para o requisito 1, decidimos implementar um gráfico linear de forma a poder analisar a evolução do índice ao longo dos dias. Fizemos o mesmo para o caso do requisito 2. Quanto aos requisitos 3, 4 e 5 optámos por representar a informação associada em formato de tabela. De notar que em *Saiku* as *queries* devem ser planeadas no separador apropriado para esse efeito e, seguidamente, guardadas na diretoria local para que, posteriormente, na etapa de conceção do *layout* do *dashboard*, possam serem utilizadas. De seguida, na Figura 40 está apresentado um *screenshot* do *dashboard* que foi implementado.

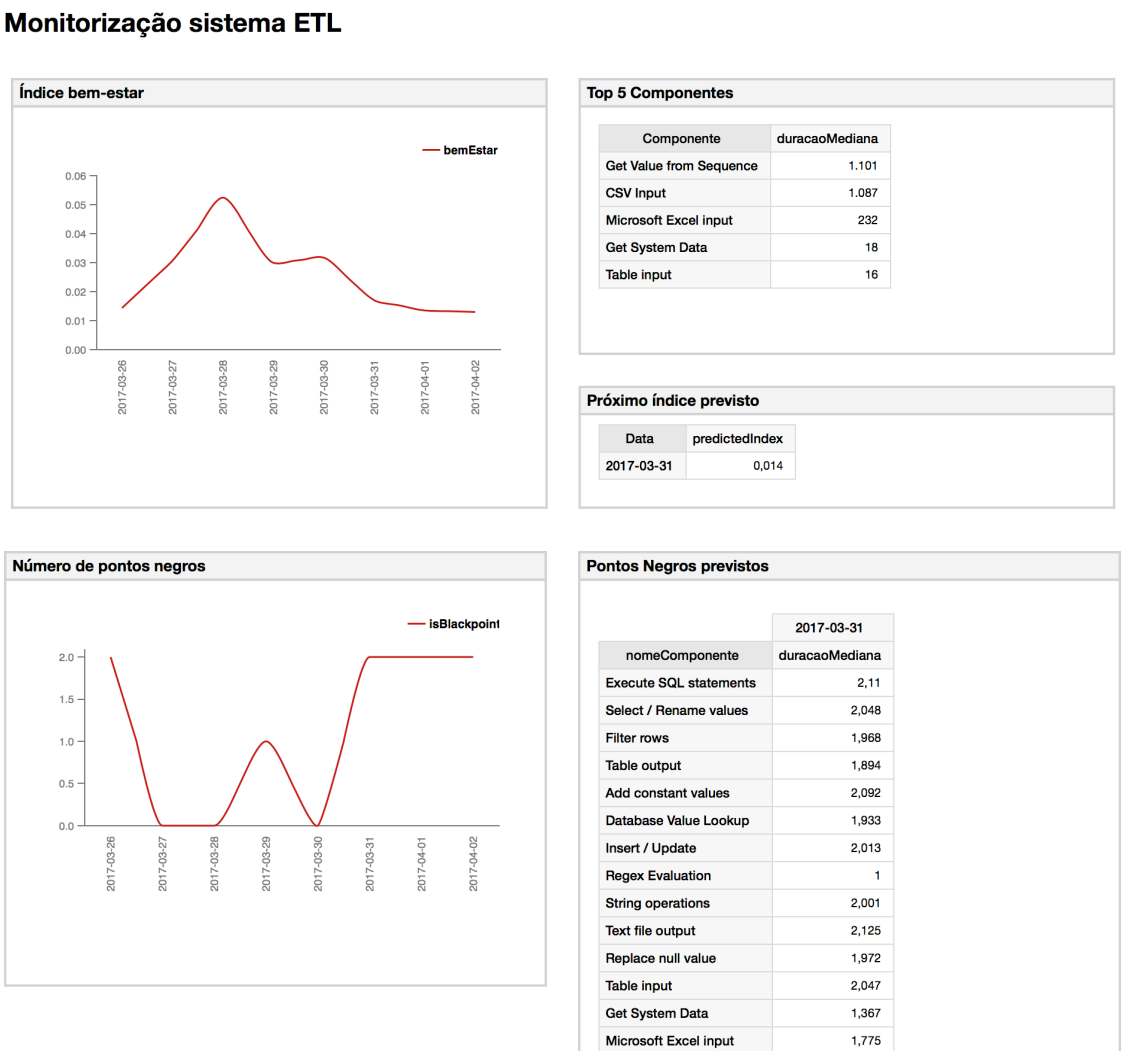


Figura 40 – Implementação do *dashboard* final.

Em termos gerais, a complexidade das *queries* utilizadas não é grande. Na base de cada um dos *widgets* que compõe o *dashboard*, excetuando a *query* que diz respeito ao índice de bem-estar, nenhuma das *queries* utilizadas implicou a manipulação de MDX. A manipulação da *interface* disponível pela ferramenta foi suficiente para alcançar os resultados necessários. A *query* relativa ao índice de bem-estar pode ser observada na Figura 41. Ao observarmos a sua composição podemos ver que, para que os resultados verificados fossem os esperados, foi necessário introduzir três membros calculados na *query,* que não fazem mais do que calcular a proporção de registos novos, a proporção ocupada da janela de oportunidade e aplicar a expressão de

normalização do resultado para o intervalo [0,1] (secção assinalada com um retângulo vermelho), respetivamente.

```
WITH
MEMBER [Measures].[nova] AS
    (([junkDim].[Junk Hierarchy ].[NrInsert].[17715])/([junkDim].[Junk Hierarchy ].[NrInsert].[17715]+[junkDim].[Junk Hierarchy 2]
.[NrUpdate].[0]+[junkDim].[Junk Hierarchy 3].[NrDelete].[0]))
MEMBER [Measures].[window] AS
    ([junkDim].[Junk Hierarchy 4].[windowSize].[1.0])
MEMBER [Measures].[bemEstar] AS
    ([Measures].[bemEstarParcial]*[Measures].[nova]*[Measures].[window] / (([Measures].[bemEstarParcial] * [Measures].[nova]  *
[Measures].[bemEstarParcial]*[Measures].[window]) + 1))
SET [~ROWS] AS
    {[DimCalendario].[Time by Month].[Data].Members}
SELECT
NON EMPTY {[Measures].[bemEstar]} ON COLUMNS,
NON EMPTY [~ROWS] ON ROWS
FROM [BlackpointDWCube]
```

Figura 41 – Implementação da *query* relativa ao cálculo do índice de bem-estar.

Capítulo 5

Conclusões e Trabalho Futuro

5.1 Comentários Finais

A dimensão e impacto da aplicação de *process mining* em processos do quotidiano não estão, ainda, exaustivamente avaliados, no entanto, a lista de trabalho na área é já digna de algum destaque. Apesar do eco de *process minng* em termos de investigação, não é fácil, tal como relatado anteriormente nesta dissertação, generalizar técnicas e métodos a aplicar. Assim sendo, iniciar um trabalho destes, nesta área, apesar de entusiasmante, torna-se facilmente assoberbante pela quantidade de informação disponível. Tudo parece indicar, no entanto, que nos próximos anos haja uma tentativa de reunir consenso nesta área, o que lhe permitirá crescer em termos de aceitação e aplicabilidade. Relativamente aos processos ETL, a sua otimização não é só naturalmente complexa, como extremamente dependente das ferramentas utilizadas na sua conceção. Com efeito, o uso de ferramentas *freeware*, apesar de adequado a ambientes académicos, traz algumas desvantagens quando se pretende recolher o registo dos passos executados pelo sistema, ao longo do tempo, por exemplo.

Relativamente ao trabalho realizado, a obtenção de dados de execução de sistemas ETL revelou-se, de facto, complicada, sobretudo porque não basta conseguir registos de execução de um processo. É necessário, sim, que esses registos se encontrem bem construídos e, além disso, que sejam em grande número, para termos uma base minimamente sólida de trabalho. Assim, apesar de termos utilizado um processo bem definido, cuja implementação é por nós bem conhecida, o volume de dados que tínhamos disponível dizia respeito à simulação de nove dias de execução do

nosso ETL, o que equivalia a cerca de 3300 registos. Ora, tal cenário revela-se algo limitado se se pretenderem generalizar os resultados obtidos à custa da nossa nova abordagem. Nesse sentido, este poderá ser apontado como um dos aspetos negativos do nosso trabalho. Outro desses aspetos poderá ser a utilização de ferramentas livres de qualquer licenciamento. De facto, a utilização de ferramentas *freeware*, apesar de suficiente em ambientes académicos e, mais particularmente, para ilustrar a nossa abordagem, limita-nos um pouco em termos de funcionalidades que, noutras ferramentas seriam *standard*, mas que nestas aplicações não o são. Um exemplo flagrante disso mesmo é o registo de *logs* de execução do processo ETL. Com efeito, o facto de não se processar de forma automática implicou a implementação de um processo alternativo que passou pela injeção manual das etiquetas temporais associadas a cada passo. Esse processo revelou-se bastante custoso, sobretudo tendo em conta o elevado número de passos distintos que tínhamos de registar. Outros aspetos menos positivos englobam a necessidade de implementar um processo alternativo para obtenção de bases de dados multidimensionais ou a estabilidade de certas versões destas ferramentas em determinados ambientes de execução. O primeiro ponto, está diretamente ligado às limitações naturais de *MySQL Workbench*. O segundo ponto diz respeito ao facto de ferramentas como *Kettle* serem implementadas em *Java*. Ora, ao ser implementado em *Java*, *Kettle* está muito sujeito a erros em tempo de execução, sobretudo se se der uma má escolha quer da versão *Kettle* que estamos a utilizar, quer da versão de *Java*. Esses problemas tendem a piorar quando o sistema operacional utilizado é *MacOS*, que é o nosso caso.

Em termos positivos destacaríamos, sobretudo, a reação e a curiosidade despertada pela nossa abordagem inovadora que alia *process mining* e sistemas ETL, algo até ao momento inédito. Nesse sentido, as nossas participações na 17ª Conferência da Associação Portuguesa de Sistemas de Informação – CAPSI'2017 - , que decorreu em Guimarães, entre os dias 6 e 7 de junho de 2017 [27], bem como na *5th World Conference on Information Systems and Technologies – WorldCist'17* – que decorreu em Porto Santo entre os dias 11 e 13 de abril de 2017 [14], vieram apenas confirmar a nossa perceção inicial de que este trabalho iria captar a atenção dos presentes. Outro aspeto positivo que gostaríamos de destacar foi o facto deste trabalho de dissertação se ter desenvolvido de forma perfeitamente calculada e linear ao longo do tempo, pelo facto de resultar de um processo bem planeado e definido logo *a priori*.

5.2 Próximos Passos

No que diz respeito a uma eventual evolução do trabalho já realizado até ao momento, poder-se-á dizer que o estado do projeto relacionado com esta dissertação está bem sustentado e estável. No entanto, até para uma melhor perceção do impacto que esta abordagem teria em ambientes reais, um próximo passo poderia passar por aplicar este método a um *data warehouse* que esteja em produção e verificar o seu comportamento relativamente à abordagem que desenvolvemos, verificando se a mesma é suficiente e se permite resolver problemas reais deste tipo de sistemas, sempre cumprindo com as premissas de não obrigar à visualização e análise de demasiados indicadores e variáveis, nem se traduzir num elevado impacto em termos de esforço inicial para colocar o sistema em funcionamento. Além disso, deveria ser testada a camada de previsão do índice de bem-estar e dos respetivos pontos negros em maior profundidade por forma a verificar se a sua acurácia tenderia a aumentar com o aumento do número de casos de treino ou se, pelo contrário, a acurácia se veria algo limitada pela variabilidade que está associada a estes sistemas. Nesse sentido, a validação da camada de previsão poderia implicar um reformular de toda a arquitetura que foi desenvolvida nesse sentido.

Bibliografia

[1] W. M. P. van der Aalst, "Extracting Event Data from Databases to Unleash Process Mining," *BPM - Driv. Innov. a Digit. World SE - 8*, pp. 105–128, 2015.

[2] W. M. P. van der A. B.F. van Dongen, "A Meta Model for Process Mining Data," *Proc. CAiSE'05 Work.*, vol. 11, no. i, pp. 309–320, 2005.

[3] B. F. A. Hompes, J. Buijs, W. M. P. van der Aalst, P. M. Dixit, and J. Buurman, "Discovering Deviating Cases and Process Variants Using Trace Clustering," *Proc. 27th Benelux Conf. Artif. Intell. (BNAIC), Novemb.*, pp. 5–6, 2015.

[4] M. Song, C. W. Günther, and W. M. P. Van Der Aalst, "Trace clustering in process mining," in *Lecture Notes in Business Information Processing*, 2009, vol. 17 LNBIP, pp. 109–120.

[5] M. Leemans and W. M. P. van der Aalst, "Discovery of Frequent Episodes in Event Logs," *Data Min. Knowl. Discov.*, vol. 1, no. 3, 1997.

[6] A. Rozinat, A. K. A. De Medeiros, C. W. Günther, A. J. M. M. Weijters, and W. M. P. Van Der Aalst, *The need for a process mining evaluation framework in research and practice: Position paper*, vol. 4928 LNCS. 2008.

[7] S. Few, *The Effective Visual Communication*. 2004.

[8] B. Torres, C. Ferreira, F. Pinto, and N. Dias, "Um Data Warehouse para um cluster de empresas - Relatório Técnico Mestrado Integrado em Engenharia Informática Universidade do Minho," Braga, Portugal, 2016.

[9] Pentaho, "Data Integration - Kettle," 2017. [Online]. Available: http://www.pentaho.com/product/data-integration.

[10] Pentaho, "Execute SQL script - Pentaho Data Integration - Pentaho Wiki." [Online]. Available: http://wiki.pentaho.com/display/EAI/Execute+SQL+script. [Accessed: 08-May-2017].

[11] Pentaho, "Block this step until steps finish - Pentaho Data Integration - Pentaho Wiki."

[Online]. Available: http://wiki.pentaho.com/display/EAI/Block+this+step+until+steps+finish. [Accessed: 08-May-2017].

[12] P. M. G. Eindhoven Technical University, "ProM Tools," 2010. [Online]. Available: http://www.promtools.org/doku.php. [Accessed: 01-May-2017].

[13] fluxicon, "Disco: Discover your processes." [Online]. Available: https://fluxicon.com/disco/. [Accessed: 15-Aug-2016].

[14] O. Belo, N. Dias, F. Pinto, and C. Ferreira, "Discovering ETL Black Points A Process Mining Approach," *5th World Conf. Inf. Syst. Technol. (WorldCIST 2017), Porto St. Island, Madeira, Port. April. 11-13*, 2017.

[15] W. Thornthwaite, "Kimball GroupDesign Tip #113 Creating, Using, and Maintaining Junk Dimensions - Kimball Group," 2009. [Online]. Available: http://www.kimballgroup.com/2009/06/design-tip-113-creating-using-and-maintaining-junk-dimensions/. [Accessed: 13-May-2017].

[16] Iconsulting, "The Dimensional Fact Model - Indyco," 2017. [Online]. Available: http://www.indyco.com/dimensional-fact-model/. [Accessed: 05-Apr-2017].

[17] M. Golfarelli and S. Rizzi, "A methodological framework for data warehouse design," *Proc. 1st ACM Int. Work. Data Warehous. Ol. - Dol. '98*, no. Dolap 98, pp. 3–9, 1998.

[18] pentaho, "Mondrian | Pentaho Community." [Online]. Available: http://community.pentaho.com/projects/mondrian/. [Accessed: 02-May-2017].

[19] C. Guyer and O. Duncan, "Querying Multidimensional Data with MDX | Microsoft Docs," 2017. [Online]. Available: https://docs.microsoft.com/en-us/sql/analysis-services/multidimensional-models/mdx/querying-multidimensional-data-with-mdx. [Accessed: 27-Apr-2017].

[20] S. Wood, "Pentaho Mondrian Documentation," 2007. [Online]. Available: http://mondrian.pentaho.com/documentation/workbench.php. [Accessed: 02-May-2017].

[21] C. W. Gunther and A. Rozinat, "Disco 1.6.0 — Flux Capacitor Median Performance Metrics," 2014. [Online]. Available: https://fluxicon.com/blog/2014/01/disco-1-6-0/. [Accessed: 13-Feb-2017].

[22] M. Guimarães, J. Saraiva, and O. Belo, "Categorização do Consumo de Energia em Sistemas de Povoamento de Data Warehouses," *5º Conferência da Assoc. Port. Sist. Informação (CAPSI'2015), Lisboa, Port. October, 2-3*, 2015.

[23] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth, "Step-

by-step data mining guide," 1999.

[24] RapidMiner, "RapidMiner Studio," 2017. [Online]. Available: https://rapidminer.com/products/studio/. [Accessed: 05-Mar-2017].

[25] Meteorite.bi, "Meteorite.bi - Home of Saiku Analytics | meteorite.bi," 2015. [Online]. Available: http://meteorite.bi/. [Accessed: 12-Apr-2017].

[26] D. Steiner, "Diethard Steiner on Business Intelligence: Mondrian 4: Get ready!," 2013. [Online]. Available: http://diethardsteiner.blogspot.pt/2013/01/mondrian-4-get-ready.html. [Accessed: 27-Apr-2017].

[27] N. Dias and O. Belo, "Determinação da Qualidade de Execução de um Sistema de Povoamento de um Data Warehouse Determining the Quality of Execution of a Data Warehousing Populating," *17.ª Conferência da Assoc. Port. Sist. Informação (CAPSI'2017), Guimarães, Port. June 6-7*, pp. 1–13, 2017.

Lista de Siglas e Acrónimos

ETL *Extract Transform Load*

PM *Process Mining*

FK *Foreign Key*

FT *Fact Table*

MDX *Multidimensional Expressions*

XML *Extensible Markup Language*

OLAP *Online Analytical Processing*

NN *Not Null*

OS *Operating System*

CSV *Comma-separated values*

SGBD Sistema de Gestão de Bases de Dados

Anexos

I. Caracterização das dimensões do cubo *BlackpointDWCube - DimCalendario*

Caracterização da dimensão					
Identificação	DimCalendario				
Descrição	Calendário do ano e seus atributos				
Tipo	Sem variação				
Crescimento	Não cresce. O povoamento desta dimensão é feito durante a fase de arranque do *Data Warehouse* para um período de 10 anos, desde a data mais antiga, i.e., 24/05/2005.				
Atributos					
Nr	**Identificação**	**Descrição**	**Domínio (Tamanho)**	**Variação [(S)im/(N)ão]**	**Exemplos**
1	Data	Data do calendário	Data	N	15/10/2013
2	Dia	Número do dia do mês	Inteiro	N	3
3	Semana	Número da semana do ano	Inteiro	N	4
4	Mes	Número do mês	Inteiro	N	7
5	Trimestre	Número do trimestre	Inteiro	N	1
6	Semestre	Número do semestre	Inteiro	N	2
7	Ano	Número do ano	Inteiro	N	2015
Hierarquia (Ramos)					
Nr	**Identificação**	**Esquema**			
1	H1	Data -> Dia -> ALL			
2	H2	Data -> Semana -> Mês -> Trimestre -> Semestre -> Ano -> ALL			
Perfis de utilização					
Administradores gerais e gestores de loja					
Observações					

Nada a assinalar

II. Caracterização das dimensões do cubo *BlackpointDWCube - DimComponente*

Caracterização da dimensão					
Identificação	DimComponente				
Descrição	Componentes utilizados na implementação do sistema				
Tipo	Sem variação				
Crescimento	Não cresce. O povoamento desta dimensão é feito durante a fase de arranque do *Data Warehouse* e não mais sofre alterações.				
Atributos					
Nr	**Identificação**	**Descrição**	**Domínio (Tamanho)**	**Variação [(S)im/(N)ão]**	**Exemplos**
1	Componente	Nome do componente	Varchar(255)	N	Execute SQL statements
2	TipoComponente	Fase do processo em que foi utilizado	Varchar(255)	N	Extração
Hierarquia (Ramos)					
Nr	**Identificação**	**Esquema**			
1	H1	Componente -> TipoComponente -> ALL			
Perfis de utilização					
Observações					
Nada a assinalar					

III. Caracterização das dimensões do cubo *BlackpointDWCube - DimCaseGrupo*

Caracterização da dimensão					
Identificação	DimCaseGrupo				
Descrição	Transformações em que o componente foi utilizado				
Tipo	Sem variação				
Crescimento	Não cresce. O povoamento desta dimensão é feito durante a fase de arranque do *Data Warehouse* e não mais sofre alterações.				
Atributos					
Nr	**Identificação**	**Descrição**	**Domínio (Tamanho)**	**Variação [(S)im/(N)ão]**	**Exemplos**
1	grupoCase	Nome do componente	Varchar(255)	N	Customer TT,Staff TT,Product TT
Hierarquia (Ramos)					
Nr	**Identificação**	**Esquema**			
1	H1	grupoCase -> ALL			
Perfis de utilização					
Observações					
Nada a assinalar					

IV. Caracterização das dimensões do cubo *BlackpointDWCube – DimProcesso_junk*

Caracterização da dimensão					
Identificação	DimProcesso_junk				
Descrição	Informações relativas ao processo ETL				
Tipo	Sem variação				
Crescimento	Um registo por cada ciclo de execução de ETL				
Atributos					
Nr	**Identificação**	**Descrição**	**Domínio (Tamanho)**	**Variação [(S)im/(N)ão]**	**Exemplos**
1	nrInsert	Número de registos marcados como “Insert”	INT	N	10
2	nrUpdate	Número de registos marcados como “Update”	INT	N	15
3	nrDelete	Número de registos marcados como “Delete”	INT	N	0
4	windowSize	Proporção da janela de oportunidade ocupada	FLOAT	N	0,96
Hierarquia (Ramos)					
Nr	**Identificação**	**Esquema**			
1	H1	nrInsert -> ALL			
2	H2	nrUpdate -> ALL			
3	H3	nrDelete -> ALL			
4	H4	windowSize -> ALL			

Perfis de utilização
Observações
Nada a assinalar

V. Caracterização das dimensões do cubo *PredictIndexCube - DimProcesso*

Caracterização da dimensão					
Identificação	DimProcesso				
Descrição	Processos que estão a ser monitorizados				
Tipo	Sem variação				
Crescimento	Não cresce. O povoamento desta dimensão é feito durante a fase de arranque do *Data Warehouse* e não mais sofre alterações.				
Atributos					
Nr	**Identificação**	**Descrição**	**Domínio (Tamanho)**	**Variação [(S)im/(N)ão]**	**Exemplos**
1	nomeProcesso	Nome identificativo do processo ETL	Varchar(255)	N	ETL Cluster de empresas
Hierarquia (Ramos)					
Nr	**Identificação**	**Esquema**			
1	H1	nomeProcesso -> ALL			
Perfis de utilização					
Observações					
Nada a assinalar					