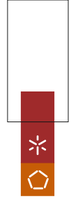


Paulo Jorge Mendes da Silva

Monitorização e Prevenção de Falhas em Bases de Dados Hospitalares

Paulo Jorge Mendes da Silva

UMinho | 2012

Universidade do Minho
Escola de Engenharia

Paulo Jorge Mendes da Silva

Monitorização e Prevenção de Falhas em Bases de Dados Hospitalares

Outubro de 2012

Universidade do Minho
Escola de Engenharia

Paulo Jorge Mendes da Silva

Monitorização e Prevenção de Falhas em Bases de Dados Hospitalares

Dissertação de Mestrado
Mestrado Integrado em Engenharia Biomédica

Trabalho realizado sob a orientação do
Professor Doutor José Manuel Machado

Outubro de 2012

Declaração

Nome: Paulo Jorge Mendes da Silva

Endereço eletrónico: silva.paulojorge@hotmail.com

Número do Bilhete de Identidade: 13382680

Título da Tese: Monitorização e Prevenção de Falhas em Bases de Dados Hospitalares

Orientador: Professor Doutor José Manuel Machado

Ano de conclusão: 2012

Designação do Mestrado: Mestrado Integrado em Engenharia Biomédica

Universidade do Minho,

Assinatura:

Agradecimentos

Esta dissertação não seria possível sem o contributo de algumas pessoas incríveis que sempre estiveram ao meu lado.

Começo por agradecer ao professor doutor José Manuel Machado pela excelente orientação que me concedeu: por sempre me incentivar e motivar; por todas as sugestões, correções, pelo tempo despendido, pela confiança que depositou em mim. Mas, acima de tudo, quero agradecer-lhe pelas críticas construtivas que permitiram um aperfeiçoamento contínuo desta dissertação.

Reservo também, um agradecimento especial ao Centro Hospitalar do Porto, local onde realizei parte da dissertação onde tive a possibilidade de contactar com profissionais de grande nível, e onde sempre me senti bem acolhido. Em particular, expresso um grande agradecimento ao doutor César Quintas, pela confiança, disponibilidade, preocupação e partilha de conhecimentos durante esta dissertação.

Um profundo obrigado ao meu colega Filipe Portela pela aprendizagem conjunta e discussão de problemas que foram surgindo aquando a minha presença no Centro Hospitalar do Porto. Obrigado pelo interesse e apoio demonstrado.

Gostaria de agradecer também à minha família que com imenso sacrifício me deu as melhores condições para o sucesso dos meus estudos, que sempre me apoiou e confiou nas minhas capacidades.

Finalmente, gostaria de agradecer aos meus amigos, que sempre me ajudaram não só ao longo da dissertação, mas ao longo da minha vida académica. Espero continuar a contar com a vossa amizade para o resto da vida. Em particular, gostaria de salientar o apoio da minha melhor amiga, Patrícia. Obrigado por seres sempre o meu porto de abrigo, é bom contar com o teu apoio incondicional. Tenho a certeza que vou continuar a ter esse apoio no futuro.

Dedico esta tese aos meus pais: à minha mãe, cujo maior orgulho era ver-me trajado e maior sonho era ver o seu filho formado, tenho a certeza que está feliz por mim; e ao meu pai, que por infortúnios nesta última etapa académica teve de se desdobrar para que eu pudesse realizar também o seu sonho: ver o filho terminar o curso.

Resumo

Atualmente, as bases de dados são consideradas ferramentas essenciais para o bom funcionamento das grandes organizações. A importância das bases de dados é ainda mais elevada nas organizações de áreas críticas, como é o caso das unidades hospitalares. Nestas unidades, as bases de dados assumem um papel vital, uma vez que armazenam uma série de informação sobre os pacientes e serviços cruciais para a prestação dos cuidados de saúde. Portanto, as bases de dados hospitalares devem ser sistemas de alta disponibilidade.

No Centro Hospitalar do Porto (CHP), encontram-se já implementados bons sistemas de tolerância a falhas, ou seja, sistemas que mantêm os dados disponíveis mesmo na presença de falhas. No entanto, estes sistemas têm um custo associado e para além disso alguns problemas continuam a acontecer. É por isso relevante o estudo de alternativas. Uma alternativa são os sistemas de previsão de falhas. Estes sistemas são já usados pelos médicos para prever o estado clínico de um paciente. Um desses sistemas é o Modified Early Warning Score (MEWS), que com base no desvio dos valores recolhidos à normalidade dos sinais vitais, usa um conjunto de *scores* para determinar o nível de risco a que o paciente está sujeito. Com esta dissertação pretendeu-se caraterizar, em termos de carga, o normal funcionamento das principais bases de dados do CHP, para que fosse possível, numa segunda fase, adaptar o sistema MEWS à realidade das bases de dados hospitalares.

Foi implementado, com sucesso, um protótipo de monitorização que, para além de calcular a gravidade da situação, permite a visualização do comportamento da base de dados através de vários dashboards e o envio de emails de alerta em casos mais graves.

Verificou-se que as principais bases de dados do CHP são bases de dados de elevada carga e utilização e que os períodos mais propícios à ocorrência de falhas são: das 10h00 às 12h00 e das 00h00 às 02h00. Para além disso, constatou-se que a adaptação da escala do MEWS à realidade das bases de dados revelou-se eficaz na deteção de situações anormais.

Abstract

Currently, the databases are considered essential tools for the proper functioning of large organizations. The importance of databases is further elevated in the organizations of critical areas, such as healthcare units. In these units, databases play a vital role, since they store a variety of information about patients and services crucial for health care. Therefore, the hospital databases should be high availability systems.

In Centro Hospitalar do Porto (CHP) are already implemented good systems for fault tolerance, i.e., systems that hold the data available even in the presence of faults. However, these systems have an associated cost and in addition some problems still occur. So, it is relevant to the study of alternatives. These systems are already used by doctors to predict the clinical status of a patient. One of those systems is the Modified Early Warning Score (MEWS), which based on the deviation of the values recorded to normal vital signs, uses a set of scores to determine the level of risk to which the patient is subject.

With this dissertation we intended to characterize, in terms of workload, the normal behavior of the main CHP databases, to make possible, in a second phase, adapt the MEWS to hospital databases reality. Was successfully implemented, a monitoring program that, in addition to calculating the gravity of the situation, allows visualization of the behavior of the database through several dashboards and sending warning emails in severe cases.

It was found that the main CHP databases have high load and use, and that the periods most favorable the occurrences of faults are: from 10h00 to 12h00 and from 00h00 to 02h00. Furthermore, it was found that the adaptation of the MEWS scale to hospital database reality has proved to be effective in detecting of abnormal situations.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

$AIDA$ Agência para a Integração, Difusão e Armazenamento da informação

$ANSI - SPARC$ American National Standards Institute - Standards Planning and Requirements Committee

ASM Automatic Storage Management

AWR Automatic Workload Repository

BI Business Intelligence

CHP Centro Hospitalar do Porto

CPU Central Processing Unit - processador

I/O Input/Output

$IGIF$ Instituo de Gestão Informática da Saúde

$MEWS$ Modified Early Warning Score

$OLAP$ Online Analytical Processing

PCE Processo Clínico Eletrónico

PGA Program Global Area

RAC Real Application Clusters

$RAID$ Redundant Array of Independent Drives

SAM Sistema de Apoio Médico

$SAPE$ Sistema de Apoio à Enfermagem

SGA System Global Area

$SGBD$ Sistema de Gestão de Base de Dados

$SONHO$ Sistema Integrado de Informação hospitalar

SQL Structured Query Language

SSH Secure Shell

Capítulo 1

Introdução

A evolução dos sistemas computacionais conduziu ao processo de informatização massiva dos serviços nas organizações das mais diversas áreas (empresas, universidades, bibliotecas, hospitais, etc.). Ao mesmo tempo, cresceu a importância da informação para as organizações, sendo que hoje em dia o sucesso ou insucesso de uma organização pode estar relacionado com a forma com que esta gere a informação. Esta conjuntura atribuiu às bases de dados o papel de ferramentas fundamentais em qualquer tipo de organização. Com a utilização de base de dados é possível proceder ao armazenamento e à gestão da gigantesca quantidade de dados presente numa organização, possibilitando desta forma um acesso simples e eficaz à informação. Estas vantagens tornam as bases de dados imprescindíveis para a realização das tarefas do dia-a-dia de uma organização.

A importância das bases de dados é ainda mais elevada em organizações que prestam serviços críticos como é o caso das unidades hospitalares. Nestas instituições já se encontram implementadas bases de dados que contêm as informações administrativas e clínicas relativas aos utentes. Para além disso, as informações armazenadas servem de suporte aos atos clínicos, aos processos organizativos e a um conjunto de aplicações responsáveis por gerir, disponibilizar e promover a partilha da informação. As bases de dados assumem assim um papel vital nas unidades hospitalares.

1.1 Motivação

Devido ao papel fundamental que as bases de dados possuem nas unidades hospitalares, estas necessitam de estar permanentemente disponíveis e em boas condições de funciona-

mento. Atualmente existem tarefas que são impensáveis de realizar sem o auxílio destas ferramentas poderosas e um pequeno período de paragem poderá trazer graves consequências para a qualidade dos serviços prestados pelas unidades hospitalares. É então necessário garantir que estas bases de dados sejam sistemas de alta disponibilidade. Para tal, no caso do Centro Hospitalar do Porto (CHP), local da realização deste estudo, já existem implementados mecanismos de tolerância a falhas em bases de dados. Estes mecanismos utilizam o processo de redundância de dados, de componentes ou de ambas em simultâneo para que seja possível, por exemplo, no caso de existir uma falha num componente, a base de dados continuar operacional. Para além disto, estes mecanismos também podem ser utilizados para balanceamento de carga e para recuperação da base de dados em caso de falha.

Não obstante destas vantagens, estes mecanismos apenas permitem a tomada de ações *a posteriori* da falha, ou seja, apenas permitem a tomada de medidas para diminuir os efeitos de uma falha, nada é feito para as evitar. Para além disto, estes sistemas implicam sempre um certo custo para a sua implementação.

Neste contexto surge a monitorização e prevenção de falhas, que consiste na definição de modelos para a monitorização de eventos e situações que possam originar falhas. Com base nestes modelos é possível a intervir com alguma antecedência tendo em vista a prevenção de falhas.

1.2 Objetivos

Os modelos de monitorização e previsão de falha são utilizados em várias áreas, até mesmo numa área crítica como é a medicina. Um exemplo destes modelos é o Modified Early Warning Score (MEWS). Este modelo parte do pressuposto que um problema grave de saúde é muitas vezes antecedido pela deterioração fisiológica. Sendo assim, procede-se à monitorização dos sinais vitais do paciente, aos quais são atribuídos valores (*scores*) em função do desvio aos valores considerados normais. Com base nesses *scores*, tenta-se perceber quando poderá ocorrer um problema grave, por exemplo a falência de um órgão [1].

Nesta dissertação pretende-se adaptar este modelo à realidade das bases de dados hospitalares do CHP. Para tal, é necessário entender as arquiteturas e caraterizar a carga

de trabalho das principais bases de dados do CHP, o que permitirá a definição do normal comportamento das mesmas. Para além disso, pretende-se implementar um modelo que monitorize as bases de dados e emita alertas caso os valores dos parâmetros não coincidam com os limites dinamicamente definidos.

1.3 Estrutura

A restante dissertação encontra-se organizada da seguinte forma:

- **Capítulo 2** - São abordados os conceitos gerais sobre base de dados, em particular das bases de dados Oracle. Conceitos estes, importantes para a compreensão da restante dissertação.
- **Capítulo 3** - É apresentado o estado da arte desta temática. São descritas as ferramentas já utilizadas no CHP no que respeita a falhas em bases de dados e são apresentados alguns estudos sobre a temática de previsão de falhas, não só, relacionados com as bases de dados mas também com a medicina.
- **Capítulo 4** - São desenvolvidos os conceitos teóricos relativos às falhas em bases de dados, procedendo-se também à explicação das técnicas e ferramentas utilizadas para a realização desta dissertação.
- **Capítulo 5** - É apresentado o caso de estudo e a metodologia seguida. A apresentação do caso de estudo foca-se na temática da evolução dos sistemas hospitalares e na definição das arquiteturas das bases de dados em análise. Em relação à metodologia, são descritos os principais passos para a realização desta dissertação.
- **Capítulo 6** - São apresentados e discutidos os resultados do estudo efetuado às duas bases de dados do CHP.
- **Capítulo 7** - É elaborada uma breve descrição das publicações a que esta dissertação deu origem.
- **Capítulo 8** - Serão descritas as principais conclusões obtidas após a realização deste estudo. Para além disso, são apresentadas algumas sugestões como trabalho futuro.

Capítulo 2

Base de dados: Abordagem geral

Hoje em dia as bases de dados assumem-se como ferramentas poderosas e essenciais para a gestão da informação numa organização. Neste capítulo, é possível ficar a conhecer um pouco mais sobre as bases de dados e sobre os seus componentes, olhando com particular atenção para o caso das bases de dados Oracle.

2.1 Conceitos

Os sistemas que garantem a manutenção, o armazenamento organizado dos dados e a segurança dos mesmos, são as **bases de dados** [2, 3]. Desta forma, pode-se definir uma base de dados como uma coleção de dados, logicamente relacionados, que são geridos por um **Sistema de Gestão de Base de Dados (SGBD)** [4]. Este SGBD é composto por um conjunto de programas responsável por armazenar, gerir, organizar e proteger os dados. Além disso, promove a partilha da informação armazenada por vários utilizadores e/ou aplicações, garantindo sempre a integridade dos dados [4, 5]. Por uma questão de simplicidade, muitas vezes é utilizado o termo base de dados para definir o conjunto: base de dados e SGBD. No entanto, quando se fala em sistemas de bases de dados é importante referir que existem outros componentes para além do SGBD, nomeadamente:

- A nível de hardware - Existe um conjunto de dispositivos físicos responsável pelo armazenamento dos dados e funcionamento da base de dados (discos, processadores, memórias, cabos de conexão, entre outros) [6].
- A nível de software - Para além do conjunto de programas pertencente ao SGBD, é comum os sistemas de bases de dados possuírem software de manipulação de dados

(por exemplo, SQL (Structured Query Language) Developer) [6].

- A nível de recursos humanos - São várias as pessoas envolvidas com os sistemas de base de dados (designer da base de dados, administrador do sistema, utilizadores) [6].

A interligação destes componentes revela-se um desafio importante para os designers e administradores principalmente se o sistema for muito complexo [2].

Com o intuito regular a implementação de sistemas de bases de dados e o relacionamento entre os seus componentes, surgiu nos anos 70, a arquitetura **ANSI-SPARC** (American National Standards Institute - Standards Planning and Requirements Committee) [6, 7]. Esta arquitetura é importante para demonstrar algumas funcionalidades do SGBD, nomeadamente na organização dos recursos. Muitas vezes é também denominada por arquitetura de 3 níveis (three-level architecture), devido a partir do princípio de que a estrutura de uma base de dados deve estar dividida em três camadas de abstração: a camada externa, conceptual e interna/física [2, 7]. Estas três camadas podem ser facilmente visualizadas através do esquema da Figura 2.1.

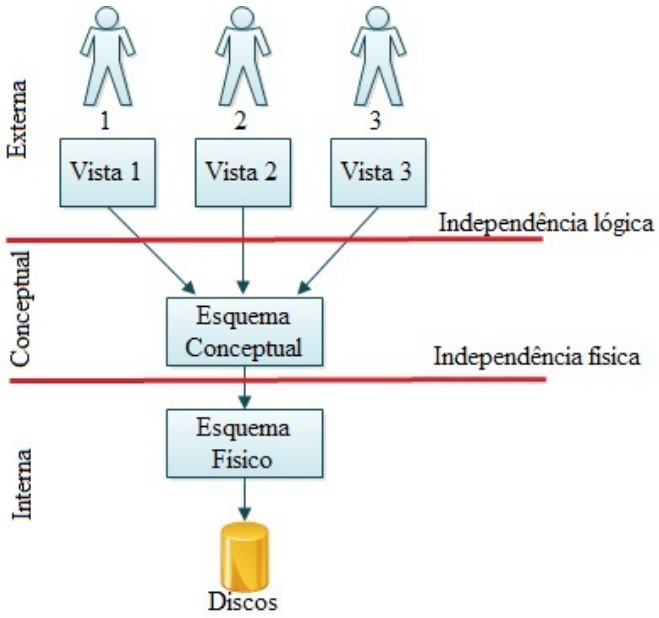


Figura 2.1: Arquitetura ANSI-SPARK (three-level)

A camada externa é a única à qual os utilizadores têm acesso e é responsável por disponibilizar um conjunto de vistas[1] adequadas ao utilizador em questão. Sendo que,

[1] São representações personalizadas dos dados, sendo que, os dados presentes nestas, derivam das tabelas de origem. Úteis para a simplificação de tabelas complexas e para questões de segurança [8].

uma vista destinada a um utilizador apenas contém os dados relevantes para este [2].

Na camada conceptual encontram-se especificados todos os dados e relações que se encontram armazenados, definindo assim um esquema lógico que serve de suporte à camada externa. É responsável também por garantir uma independência entre a camada externa e a camada interna [2, 7].

A camada interna está relacionada com a estrutura física da base de dados. Nesta camada, são definidas as estruturas e os ficheiros onde são armazenados os dados. Para além disso, são especificadas as ligações com o sistema operativo (alocação de memória, alocação de espaço, etc.) [2].

O objetivo principal desta divisão é promover a independência física e lógica dos dados. Sendo que, a independência física, como se pode observar na Figura 2.1, verifica-se entre a camada interna e conceptual. Esta independência permite aos administradores do sistema alterar a estrutura física da base de dados (mudar o diretório do ficheiro físico, ou mudar o disco) sem que os utilizadores se apercebam da mudança [7].

Em relação à independência lógica, esta verifica-se entre a camada conceptual e a externa, permitindo adicionar objetos à base de dados sem alterar as vistas dos utilizadores que não contêm dados passiveis de ser alterados por essa operação [2].

Através desta divisão o SGBD consegue gerir de uma forma mais eficaz e organizada, as informações contidas na base de dados e os diversos pedidos elaborados pelos utilizadores garantindo a satisfação dos mesmos [2].

2.2 Modelos de base de dados

Os SGBD podem ser classificados de acordo com o modelo que utilizam para armazenar os dados numa base de dados e a forma como os interligam. Os modelos devem permitir organizar os dados de uma forma clara e intuitiva, para que seja possível efetuar uma gestão eficaz dos dados armazenados [3].

O modelo mais utilizado é o modelo relacional desenvolvido por Edgar Codd [5, 7]. Neste modelo, os dados são agrupados por entidades sendo armazenados em tabelas que representam essas mesmas entidades. As tabelas são constituídas por colunas (que determinam o tipo de dados) e por linhas (onde se encontram os campos com os respetivos valores). Várias tabelas podem ser relacionadas através dos campos que possuem em

comum, aumentado assim a capacidade de armazenamento (eliminando dados redundantes) e tornando mais rápido os processos de armazenamento/recuperação. Os campos que permitem identificar univocamente uma linha são denominados de chave primária, sendo normalmente os utilizados para estabelecer as relações entre tabelas. Se numa tabela existir um campo com os mesmos atributos de uma chave primária de outra tabela, então esse campo é denominado por chave estrangeira, pode-se então estabelecer uma relação entre as duas tabelas [3, 5, 7].

Com a evolução das linguagens de programação orientadas aos objetos e com o aumento da diversidade do tipo de dados a armazenar têm surgido propostas para dotar o modelo relacional de ferramentas que permitam trabalhar com linguagens orientadas a objetos. Surgem assim as bases de dados *object-relacional (O-R)* que visam utilizar as mais valias do modelo relacional e introduzir características orientadas a objetos que permitem responder às necessidades de hoje em dia [3].

2.3 Segurança

O aumento da quantidade e qualidade da informação presente nas organizações torna a segurança um aspeto fundamental em qualquer sistema de informação. A segurança deve ser implementada em todos os constituintes do sistema de informação, desde o sistema operativo, base de dados e rede onde os dados circulam [3]. Relativamente à segurança das bases de dados, principalmente nas grandes organizações, esta assume-se como uma questão prioritária. A quantidade de informação armazenada, o número elevado de utilizadores em simultâneo e a complexidade das arquiteturas, implica realizar um grande esforço para garantir a segurança e o bom desempenho das bases de dados. Uma base dados segura deve possuir três características fundamentais [3, 9]:

- **Confidencialidade** - implementada através de mecanismos de prevenção de intrusos, evitando que pessoas não autorizadas acedam e divulguem os dados armazenados. Cada utilizador deve possuir um conjunto de regras sobre a que informações pode aceder ou não, protegendo desta forma a sua privacidade e também a dos demais utilizadores. O acesso à informação deve estar de acordo com a legislação em vigor e com as regras da instituição onde a base de dados está implementada [3, 9, 10].

- **Integridade** - garantida através de mecanismos que impedem a modificação indevida dos dados. Estes mecanismos para além de verificar se um determinado utilizador tem privilégios para modificar os dados, verificam também se essa mudança é válida, ou seja, se alterações dos dados respeitam as regras previamente definidas. Desta forma, é possível manter a informação da base de dados incorrupta, inviolável e consistente [3, 9, 10].
- **Disponibilidade** - assegurada através de mecanismos de recuperação de falhas causadas por intrusos e de mecanismos que promovem a tolerância a falhas de componentes. Estes mecanismos permitem o sistema continuar a responder em tempo útil aos pedidos dos utilizadores não comprometendo o normal funcionamento da organização [9].

Normalmente os mecanismos mencionados acima fazem parte dos SGBD. Uma vez que esta dissertação centraliza-se na temática da disponibilidade das bases de dados, esta será abordada com maior detalhe na próxima secção.

2.4 Disponibilidade

Atualmente, nas organizações mais modernas, as bases de dados revelam-se imprescindíveis para a realização das tarefas do dia-a-dia, sendo utilizadas 24 horas por dia, 7 dias por semana com o intuito de garantir o acesso rápido e eficaz à informação pretendida pelos utilizadores. Para além disso, perdas de informação devido a falhas nas bases de dados são cada vez mais intoleráveis. Sendo assim, promover uma elevada disponibilidade destes sistemas é uma questão essencial para o bom desempenho das organização [11, 12].

No entanto, a questão da disponibilidade das bases de dados é bastante complexa. Em primeiro lugar, a disponibilidade é medida através da perceção dos utilizadores finais, ou seja, normalmente o utilizador final (que por norma é bastante exigente) é quem sentencia se a base de dados está disponível ou não. Além disso, uma base de dados pode estar ligada e o sistema estar indisponível [13]. Várias situações podem tornar o sistema de base de dados indisponível apesar de ele estar ligado, nomeadamente [14]:

- O sistema pode estar inacessível devido a problemas de rede ou vírus que fecham a entrada aos utilizadores [14].

- O sistema pode estar demasiado lento, impedindo que os utilizadores realizem as tarefas pretendidas. A lentidão do sistema pode estar relacionada com a falta de recursos ou elevado número de operações. Estas situações podem levar à falha do sistema [14].
- O sistema pode apresentar falhas intermitentes, isto faz com que os utilizadores percam a confiança no sistema, decidindo considerá-lo indisponível [14].

Por estas razões, a disponibilidade de um sistema de bases de dados está dependente de três fatores. É necessário que a base de dados esteja ligada, que seja possível realizar um acesso eficaz aos dados e que todos os elementos da base de dados (rede, espaço em disco, memória, processador (CPU), etc.) estejam disponíveis. Só assim é possível a realização dos pedidos dos utilizadores num tempo aceitável [13, 14].

Em termos matemáticos a disponibilidade pode ser obtida através da fórmula [12, 14]:

$$A = MTTF/(MTTF + MTTR) \tag{2.4.1}$$

em que:

- A (Availability) - representa a disponibilidade [12].
- MTTF (Mean Time to Failure) - Média do espaço temporal onde não ocorrem falhas [12].
- MTTR (Mean Time to Repair) - Média do espaço temporal que é necessário para reparar/recuperar o sistema após uma falha [12].

Devido ao facto de as bases de dados hoje em dia serem utilizadas em muitas áreas críticas, como no caso particular das bases de dados hospitalares, é importante garantir que o valor de A esteja perto dos 99,999%, ou seja, que as bases de dados sejam sistemas de alta disponibilidade. Para assegurar esta alta disponibilidade, os sistemas de base de dados implementados devem possuir [15]:

- Hardware e software de **confiança**. Apesar dos dois componentes serem importantes, o software assume o papel mais relevante, uma vez que, uma boa solução de software pode disfarçar o uso de hardware de menor qualidade, permitindo à base de dados continuar em funcionamento apesar de uma falha de hardware. O Oracle Real Application Clusters (RAC), que será abordado no próximo capítulo, é um exemplo de uma boa solução de software. [15].
- Mecanismos de **recuperação**, para que na presença de falhas seja possível conhe-

cer rapidamente quais as falhas que ocorreram e quais os passos a tomar para a recuperação do sistema [15].

- Mecanismos que permitam uma **rápida deteção da falha**. É importante monitorizar os componentes do sistema, para que perante a ocorrência de uma falha não seja necessário perder tempo a localiza-la, mas sim em tentar recuperar da mesma [15].
- Mecanismos que garantam **operações contínuas**, para que por exemplo, existindo necessidade de fazer a manutenção de algum equipamento esta seja transparente para o utilizador da base de dados [15].

Possuindo estas quatro principais características pode-se afirmar que a base de dados, é de facto, um sistema de alta disponibilidade e está apta para um funcionamento contínuo, mesmo na presença de falhas.

2.5 Base de dados Oracle

Atualmente, os SGBD da Oracle são dos sistemas comerciais de base de dados mais populares e mais utilizados em todo o mundo, sendo disponibilizados em várias versões cada uma delas com características e funcionalidades próprias. Em virtude dos principais sistemas de bases de dados utilizados no CHP serem soluções Oracle revela-se importante apresentar as características fundamentais dos mesmos.

Um conceito extremamente importante nas bases de dados Oracle é o conceito de **instância**. Embora o termo **base de dados** seja utilizado muitas vezes para mencionar de igual forma os dois conceitos, estes não são iguais. Uma instância é um conjunto de processos e memória que permitem aceder e gerir os dados armazenados na base de dados. Uma instância apenas pode aceder a uma base de dados, no entanto, uma base de dados pode possuir várias instâncias sendo esta a base dos sistemas de *clustering*. Por sua vez, a base de dados corresponde às estruturas responsáveis pelos armazenamento dos dados [3, 8, 16].

Nas bases de dados Oracle existem estruturas lógicas e estruturas físicas que se encontram separadas propositadamente para que seja possível gerir o armazenamento físico dos dados sem interferir com o acesso às estruturas lógicas [8]. Os dados são então divididos em estruturas lógicas denominadas de **tablespaces**. Cada **tablespace** é constituída por

um ou mais ficheiros denominados de **datafiles**, sendo que estes apenas podem pertencer a um e só um **tablespace** [3]. Para além dos **datafiles**, existem outras estruturas físicas de grande importância:

- **Control File** - Neste ficheiro é armazenada a totalidade da informação necessária para o bom funcionamento da instância da base de dados. Contém informação sobre o nome da base de dados, o nome e a localização dos **tablespaces, datafiles e dos ficheiros redo log** [3, 8].
- **Redo Log** - Nestes ficheiros é efetuado o registo de todas as transações efetuadas na base de dados. Através destes ficheiros é possível recuperar os dados perdidos que ainda não tinham sido escritos nos **datafiles** aquando uma existência de uma falha [3, 8].

Relativamente à estrutura de memória dos sistemas Oracle, é conveniente proceder à explicação de alguns conceitos, para melhor compreender o seu funcionamento. Um desses conceitos é a **System Global Area (SGA)**. A SGA consiste num espaço de memória partilhada que permite armazenar temporariamente dados provenientes da base de dados. Para além disso, possibilita a partilha de informação útil para os diferentes processos necessários para o funcionamento da instância. A memória é alocada no início da instância e libertada quando esta termina, sendo que o seu conteúdo é alterado segundo o algoritmo LRU (Least Recently Used), ou seja são substituídos os dados que foram acedidos à mais tempo. A SGA pode ser dividida em diferentes estruturas [3, 8]:

- **Buffer Cache** - Ocupa a maior parte da SGA. Contém os últimos blocos de dados utilizados, permitindo desta forma um acesso mais rápido aos dados pretendidos, pois é necessário ir consultar-los de novo a disco [8].
- **Redo Log Buffer** - Armazena as alterações efetuadas à base de dados, posteriormente estas informações são escritas nos ficheiros de **redo log** [8].
- **Shared Pool** - Armazena informação sobre objetos usados frequentemente. É constituída por três zonas distintas: *library cache* - quando se executa uma operação SQL ela passa por várias etapas, algumas informações ficam armazenadas aqui para que quando for necessário executar o mesmo SQL este seja mais rápido; *dictionary cache* - contém informações sobre objetos e respetivas permissões; *shared SQL areas* - mantém os dados resultantes de uma instrução SQL em memória, apenas para dados estáticos [8].

Para além da SGA, existe uma área de memória denominada de **Program Global Area (PGA)** que contém informação relativa a cada processo a que está associada, ou seja, é uma memória não partilhada. A cada sessão está associada uma PGA que contém [8]:

- **Memória de sessão** - Informações relativas às definições e variáveis associadas a cada sessão (informação sobre *login*, formato de visualização de datas, etc.) [8].
- **SQl privado** - Informações utilizadas em queries complexas (ordenações, junções), tais como, valores associados a variáveis durante a execução de uma instrução SQL. Para além disso, nesta zona são também armazenados os cursores [8].

Relativamente ao processos, nos sistemas Oracle é efetuada a distinção entre processos de utilizador, processos de servidor e processos de *background*. Os processos de utilizador estão associados a cada conexão à base de dados, tendo associado a eles um processo de servidor e a respetiva PGA. Por sua vez, os processos do servidor são utilizados para dar resposta aos pedidos dos utilizadores. Por último, os processos de *background*, são processos autónomos com funções específicas, salientando-se os seguintes [2]:

- **Database writer (DBWR)** - é um processo criado aquando a inicialização da instância. É responsável por escrever o conteúdo dos *buffers* no disco. Em primeiro lugar são escritos os blocos acedidos à mais tempo, de forma a manter em memória os blocos mais requisitados para um acesso mais rápido à informação pretendida. O processo de escrita é realizado quando não existe memória suficiente para carregar novos blocos, e também periodicamente garantindo assim a segurança da informação em caso de falha da memória [3].
- **Log writer (LGWR)** - processo responsável pela escrita da informação armazenada no *buffer redo log* nos ficheiros de *redo log*. Após um utilizador marcar uma transação com *commit*, as informações associadas a essa transação são escritas logo em ficheiro [2, 3].
- **System Monitor (SMON)** - é o processo responsável pela monitorização e recuperação da instância se necessário. Periodicamente verifica se está tudo bem com a instância e se necessário procede à recuperação da mesma usando as informações armazenadas nos ficheiros *redo log* [3].
- **Process Monitor (PMON)** - processo responsável por libertar os recursos do sistema em caso de falha dos processos dos utilizadores [3].

- **Archiver (ARC)** - Permite a criação de ficheiros de *backup* para garantir a segurança da informação em caso de problemas com o ficheiro de *redo log* [2].

Existe ainda um processo especial denominado de ***listener***. Este processo corre no servidor da base de dados e é responsável por receber os pedidos de conexão dos utilizadores e por os encaminhar para o servidor da base de dados desejado.

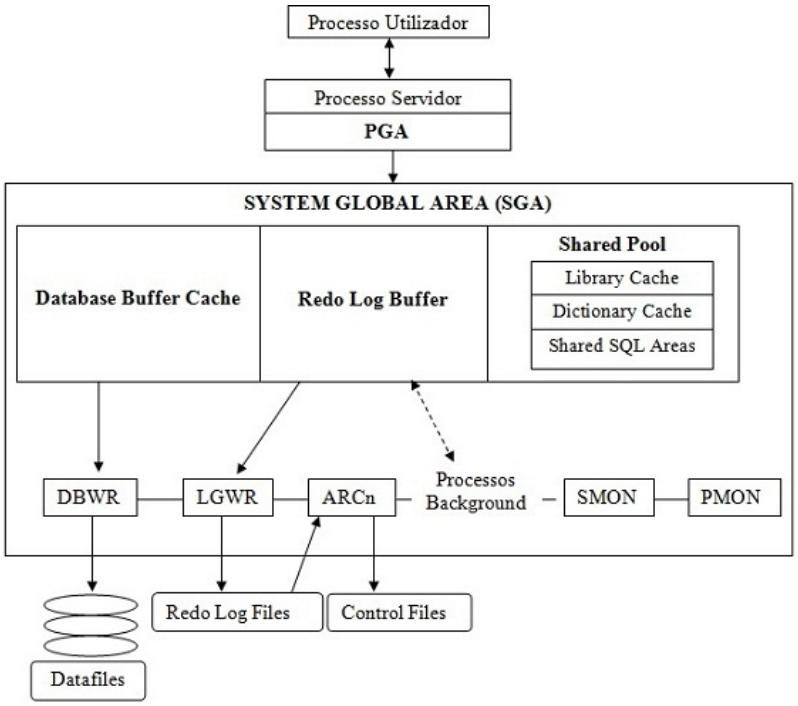


Figura 2.2: Esboço da arquitetura da base de dados Oracle

Na Figura 2.2, encontra-se um esboço da arquitetura das bases de dados Oracle, onde se pode observar a ligação entre os diferentes constituintes acima explicados.

Capítulo 3

Estado da arte

Ao longo dos anos, tem-se verificado um grande aumento de utilização das bases de dados por organizações das mais diversas áreas. Muitas organizações estão dependentes das suas bases de dados para executar tarefas do dia-a-dia, por isso, necessitam que estas sejam sistemas de alta disponibilidade. Esta necessidade levou ao aparecimento de alguns sistemas de tolerância a falhas que têm por base mecanismos de redundância [17, 18].

3.1 Sistema de tolerância a falhas

Com vista a proceder ao incremento da disponibilidade podem ser utilizados mecanismos de tolerância a falhas com base na redundância de dados, processos e ou equipamentos. Para além de aumentar a disponibilidade, a utilização destes mecanismos oferece outras vantagens a nível de performance e redução de carga, por exemplo, numa altura de elevada carga os utilizadores podem ser divididos pelos vários servidores aumentando a performance global do sistema. Estes sistemas são também sistemas bastante úteis para a recuperação de falhas [2, 17].

A redundância de dados pode ser implementada através de dois níveis: o nível físico e o nível lógico [17, 18]. Relativamente ao **nível físico**, pretende-se implementar a redundância de dados em distintos componentes físicos. No entanto, é necessário garantir que este mecanismo é transparente para a base de dados, ou seja, apesar de existem várias cópias dos dados em vários dispositivos, a base de dados trabalha "apenas" com uma cópia. Este mecanismo pode ser implementado utilizando por exemplo a técnica RAID (**Redundant Array of Independent Drives**) [17].

Esta técnica consiste no armazenamento dos dados num conjunto de discos redundantes e independentes que são geridos por um controlador promovendo a segurança e melhorando o desempenho [19]. Existem vários níveis de RAID, sendo que a escolha do mais apropriado depende da base de dados em questão. Em sistemas de bases de dados Oracle, o nível mais aconselhado é o nível RAID 10. Neste nível encontram-se dois discos espelhados, ou seja, um é a cópia do outro, sendo que este par é posteriormente distribuído para um novo par de discos promovendo assim a redundância de dados. Preferencialmente, os componentes físicos devem estar em locais geograficamente afastados para que seja possível a recuperação caso exista um incêndio ou uma catástrofe natural num dos pontos [20][21].

A **nível lógico**, a replicação consiste em manter expressamente várias cópias dos dados criando bases de dados *standby*. Quando ocorre uma mudança na base de dados principal essa mudança deve também refletir-se nas restantes cópias, a este processo dá-se o nome de replicação [2, 17]. Dependendo da altura da atualização a replicação pode ser dividida em duas categorias [2]:

- **replicação síncrona** - quando a atualização das bases de dados *standby* ocorre imediatamente após às alterações na base de dados principal [2].
- **replicação assíncrona** - quando a atualização das bases de dados *standby* é realizada algum tempo após às alterações na base de dados principal [2].

Estas duas categorias apresentam simultaneamente vantagens e desvantagens pelo que a escolha da categoria a utilizar depende do tipo de dados armazenados na base de dados e das organizações em questão [2].

Existem vários mecanismos para a implementação de redundância com vista a aumentar os níveis de disponibilidade das bases de dados, no entanto, neste trabalho serão apenas abordados os mecanismos utilizados no CHP, nomeadamente: **Real Application Cluster (RAC)** e o **Data Guard**.

3.1.1 Real Application Cluster

O mecanismo RAC é disponibilizado pela Oracle, sendo o seu objetivo elevar a disponibilidade e escalabilidade das bases de dados. Este objetivo é atingido recorrendo à arquitetura física da Figura 3.1. Esta arquitetura consiste numa base de dados partilhada à qual se pode aceder a partir de qualquer servidor/computador, no qual esteja insta-

lada uma instância da base de dados e uma instância do Automatic Storage Management (ASM)[1][8, 11, 16].

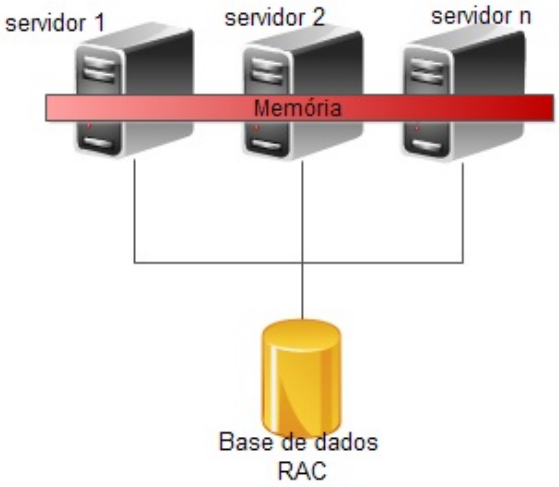


Figura 3.1: Arquitetura RAC

É importante salientar que estes servidores possuem uma memória cache, a qual pode ser acedida por outro qualquer servidor do *cluster*, através do mecanismo de **cache fusion**. Este mecanismo permite que um servidor possa aceder a um bloco que esteja na cache de outro servidor ao invés de ter de ir buscar o bloco a disco diminuindo assim o tempo de resposta. A coerência dos dados presentes na cache global (conjunto das n caches individuais) é assegurada por outro mecanismo da Oracle, o **Global Cache Service**, através, por exemplo, do bloqueio de blocos que estejam a ser modificados num determinado instante [16].

O RAC é bastante útil, uma vez que se ocorrer uma falha num servidor a base de dados estará disponível através do acesso pelos outros servidores. Outra vantagem, é o facto de permitir o balanceamento de carga, ou seja, no caso de existir um pico de carga, o mecanismo pode-se adaptar e dividir os utilizadores pelos servidores, diminuindo assim a carga individual de cada um deles [11]. Existem dois tipos de balanceamento de carga [16]:

- **Lado do Cliente** - O balanceamento de carga é efetuado através dos *listeners*. O cliente é associado a um dos nodos, com a desvantagem de uma nova sessão poder ser associada ao nodo que está com mais carga [16].
- **Lado do Servidor** - O listener redireciona a sessão para o melhor nodo da instância

[1]Ferramenta com uma interface simples para gestão dos dados armazenados.

naquele momento usando um mecanismo de aconselhamento de distribuição de carga [16].

O RAC permite realizar manutenção a um servidor sem comprometer a ligação à base de dados, pois esta continua disponível a partir dos outros servidores. Para além disso, derivado à grande escalabilidade deste mecanismo é possível aumentar o desempenho do sistema, de uma forma muito simples, adicionando novos servidores com acesso à base de dados [8, 16].

A gestão das instâncias, presentes na arquitetura RAC, é efetuada por um programa denominado por **Oracle Clusterware**. Este programa é responsável por monitorizar todos os componentes da arquitetura RAC e em caso de falha providência os mecanismos necessários para manter o sistema disponível [11, 16].

Para além desta arquitetura padrão, existe outro tipo de ambiente RAC: o RAC estendido, Figura 3.2. Esta topologia consiste em ter dois sistemas RAC em locais geograficamente diferentes, ou seja, existem duas bases de dados RAC às quais podem aceder todos os servidores que respeitem as condições já mencionadas acima [11, 16].

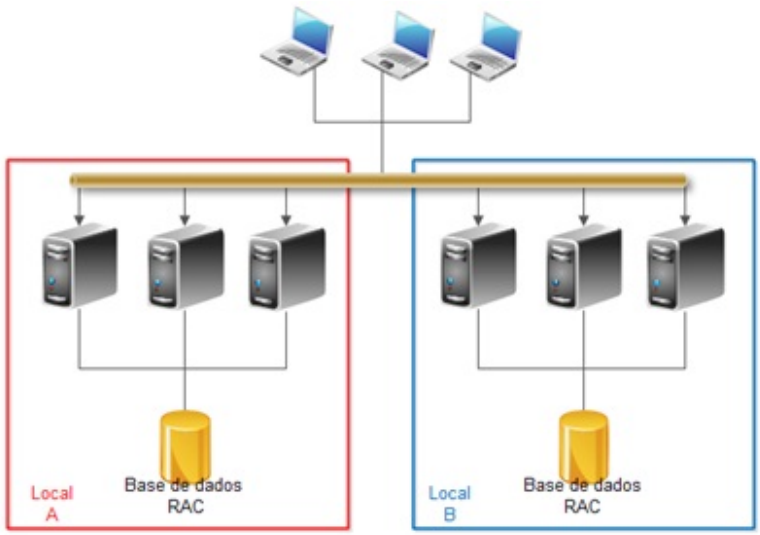


Figura 3.2: Arquitetura RAC estendido

Em caso de desastre natural ou incêndio num dos locais, a base de dados continua disponível através da cópia que é mantida noutro local [16].

Tendo em vista a coerência dos dados nas duas bases de dados, a ligação entre as duas deve ser feita através de fibra ótica de modo a aumentar a velocidade das comunicações [11].

3.1.2 Data Guard

Outra das ferramentas disponibilizadas pela Oracle para promover sistemas de alta disponibilidade é o Data Guard. Este mecanismo disponibiliza um conjunto de serviços essenciais que promovem a criação e manutenção de uma ou mais bases de dados *standby* (réplicas da base de dados original) que devem estar em espaços geográficos distintos. Assim, se por alguma razão a base de dados principal se encontrar inacessível pode-se manter o sistema a funcionar recorrendo a uma das bases de dados *standby* [15]. Estas bases de dados podem ser de dois tipos:

- **Físicas** - São mantidas sincronizadas com a base de dados principal através de um processo denominado por **Redo Apply**. As duas bases de dados ficam exatamente com a mesma estrutura física, sendo que, a atualização da base de dados *standby* é feita tendo por base os ficheiros *redo log* provenientes das operações aplicadas à base de dados principal [8, 15].
- **Lógicas** - Onde a sincronização é obtida através de uma técnica denominada por **SQL Apply**. A estrutura lógica das duas bases de dados é a mesma, no entanto, podem ter configurações físicas distintas. A replicação dos dados é efetuada através da conversão dos ficheiros *redo log*, da base de dados principal, em *scripts* SQL que são posteriormente executados na base de dados *standby*. A grande vantagem deste tipo de base de dados é que permite acrescentar novas tabelas e vistas possibilitando assim às organizações tirar mais partido da base de dados *standby*, podendo esta ser utilizada, por exemplo, para fins de monitorização e *reporting* [8, 15].

3.1.3 Arquitetura de máxima disponibilidade

Apesar das vantagens que as arquiteturas RAC e Data Guard possuem, estas individualmente não são suficientes para garantir o máximo de disponibilidade possível. Para que tal aconteça é necessário utilizar uma arquitetura que promova a junção do RAC com o Data Guard, como se pode observar na Figura 3.3 [15].

Esta arquitetura encontra-se dividida em dois locais: no primário encontra-se presente a arquitetura RAC e no secundário encontra-se a base de dados *standby* sendo que esta também possui uma configuração RAC. Quando um cliente realiza um pedido, esse pedido é encaminhado para o nó da base de dados que está disponível, preferencialmente no

local primário [11].

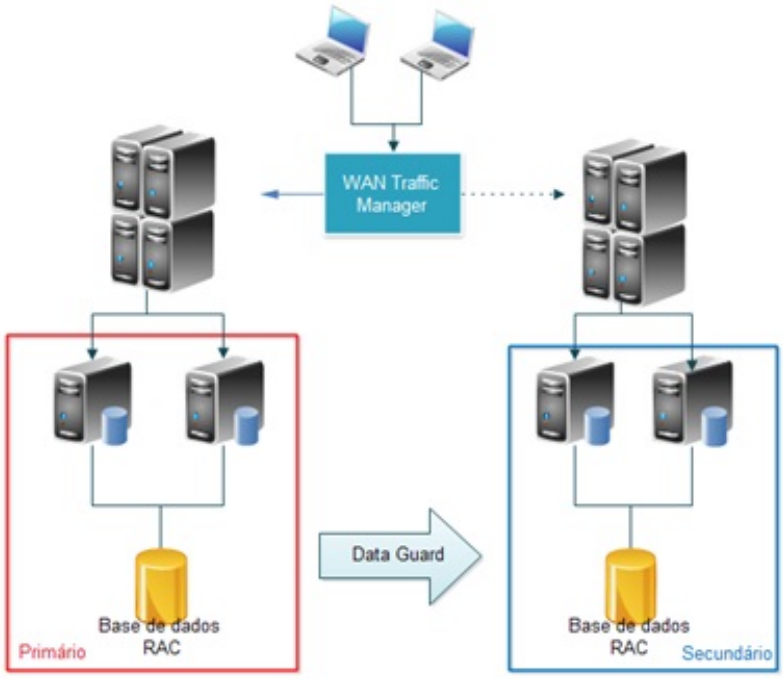


Figura 3.3: Arquitetura RAC + Data Guard

Desta forma, a base de dados fica protegida contra os vários tipos de falha, desde a falha de componentes físicos até às causadas por incidentes no ambiente que a rodeia.

Apesar das enumeras vantagens, os mecanismos de tolerância a falhas apresentam duas limitações. Uma delas é o custo dos sistemas para promover a redundância física dos dados, estes podem facilmente tornar-se em mecanismos caros ao mesmo tempo que a complexidade da base de dados aumenta [18]. A limitação com mais impacto prende-se com o facto destes sistemas focarem-se apenas nos efeitos da falha e numa ação *a posteriori* e não nas suas origens, não possuindo funcionalidades que permitam detetar e evitar precocemente falhas. Uma alternativa a estes sistemas são os modelos de previsão de falhas.

3.2 Modelo de previsão de falha (MEWS)

Os modelos de previsão de falha visam detetar com alguma antecedência falhas ou situações que possam originar falhas. Alguns modelos de previsão são já utilizados em áreas críticas, como é o caso da medicina. Na medicina é vital o uso de sistemas que permitam a previsão das condições de saúde dos pacientes, de forma a auxiliar os médicos na

tomada de decisões tornando mais rápido a prestação dos cuidados de saúde adequados. A importância destes sistemas é incrementada nos serviços de cuidados intensivos, onde a previsão do estado de um paciente pode muitas vezes salvar-lhe a vida [22].

Um exemplo destes sistemas de previsão é o MEWS (Modified Early Warning Score). Este modelo parte do pressuposto que um problema grave de saúde é muitas vezes antecedido pela deterioração fisiológica. Sendo assim, procede-se à monitorização dos sinais vitais de um paciente, aos quais são atribuídos valores (*scores*) em função do desvio aos valores considerados normais. Posteriormente, é efetuada a soma dos *scores* para a determinação do nível de risco, tentando-se perceber quando poderá ocorrer um problema grave, por exemplo a falência de um órgão [1, 23, 24].

O conceito de Early Warning Scoring surgiu pela primeira vez em 1997, sendo que nessa altura os parâmetros considerados importantes para avaliar os distúrbios funcionais eram [23]:

- **Temperatura** - Corresponde ao grau de calor que o corpo humano possui, é medida através de termómetros e o seu valor normal é 37ºC [25].
- **Frequência cardíaca** - Medida da atividade cardíaca, normalmente é expressa em batimentos por minuto [25].
- **Pressão arterial sistólica** - Corresponde ao valor mais elevado de pressão arterial de um ciclo cardíaco, ocorre aquando à sístole do ventrículo esquerdo do coração [25].
- **Estado da consciência** - Corresponde a um teste neurológico que é efetuado ao paciente de forma a avaliar a sua resposta a uma série de fatores como voz e dor [26].
- **Frequência respiratória** - Número de movimentos indicativos de respiração e expiração por unidade de tempo, pode ser determinada pelo número de vezes que o peito sobe ou desce por minuto [25].

No entanto, alguns anos mais tarde, após a realização de mais estudos, verificou-se que seriam necessários mais parâmetros para melhorar este modelo de previsão. A partir de 1999 o MEWS, para além dos sinais vitais que anteriormente monitorizava passou a avaliar também [23, 27, 28]:

- **Output urinário** - Medição do output urinário, a diminuição do output urinário pode significar um problema grave de saúde [25, 27].

- **Oxigénio no Sangue** - Procedimento que consiste na medição da concentração de oxigénio no sangue, o nível de concentração tem influência no bom funcionamento do coração e dos pulmões [25].

Todos os parâmetros monitorizados, bem como os scores associados a cada gama de valores podem ser visualizados na Tabela 3.1.

Tabela 3.1: MEWS Scores

MEWS SCORE	**3**	**2**	**1**	**0**	**1**	**2**	**3**
Temperatura (C)		< 35	35 a 36	36 a 38	38 a 38,5	> 38,6	
Frequência Cardíaca (BPM)		<40	41 a 50	51 a 100	101 a 110	111 a 130	> 131
Pressão arterial sistólica (mmHg)	<70	71 a 80	81 a 100	101 a 199		>200	
Frequência Respiratória (m-1)		<8		9 a 14	15 a 20	21 a 29	>30
Quantidade de oxigénio no sangue (%)	<85	85 a 89	90 a 93	>94			
Output urinário (ml/kg/h)	0	<0,5					
Estado da consciência		Agitação Confusão		Alerta	Responde à Voz	Responde à Dor	Não responde

O registo dos sinais vitais de cada paciente deve ser contínuo e cada medição deve ser armazenada para que se possa fazer uma análise do histórico do doente através da comparação entre os valores de medições efetuadas em instantes temporais diferentes [26, 27].

Normalmente quando algum dos parâmetros possui o *score* de 2 esse paciente deve ser alvo de uma observação frequente. No caso do score total ser igual a 4, ou então existir uma subida de 2 valores, o paciente requer atenção médica urgente. Numa situação mais extrema onde score obtido é superior a 4, o paciente corre risco de vida [1, 26].

A utilização desta escala traduz-se em muitas vantagens no processo de monitorização e prevenção de complicações de saúde nos pacientes, salientando-se as seguintes:

- Permite definir prioridades em relação às intervenções a realizar [23].
- Melhora a monitorização e observação dos pacientes uma vez que fornecem boas indicações de tendências fisiológicas [26].
- Serve de suporte à decisão médica, pois esta deixa de ser puramente subjetiva e passa a ter por base critérios quantitativos [27].

- Permite prever situações em que o paciente necessitará de internamento nos cuidados intensivos [26].

Atualmente este sistema é utilizado em hospitais de alguns países (Austrália, Reino Unido e Espanha). Em Portugal, o seu uso é muito estando já implementado no hospital do barlavento algarvio e em estudos de investigação no CHP [23, 29].

3.3 Previsão de falhas em base de dados

Embora na literatura recente já se encontrem estudos sobre previsão de performance das bases de dados e previsão de carga de trabalho (*workload*): em [30] é apresentada uma ferramenta utilizada na fase de design da base de dados, cujo o principal objetivo é simular várias situações numa base de dados, de forma a prever a performance da mesma perante essas situações, permitindo aos designers modificar a arquitetura da base de dados para obter uma melhor performance; em [13] são apresentados os desafios encontrados para a previsão de performance em bases de dados Oracle. Propõe um método para a recolha de estatísticas relacionadas com a performance e são explicados dois modelos para a previsão: o modelo baseado em rácios e o modelo matemático de regressão linear; em [31] são utilizadas técnicas de amostragem e regressão linear para avaliar e prever a performance em base de dados que são acedidas concorrentemente, ou seja, recebem mais de um pedido em simultâneo.

Todavia, não é fácil encontrar estudos sobre previsão e prevenção de falhas em bases de dados. Sendo que, em [32] encontra-se uma análise e resumo interessante de vários estudos relativos à previsão de falhas em sistemas computacionais. No entanto, estes focam-se apenas em componentes e/ou em métricas específicas, o que é algo limitativo para analisar um sistema complexo como é a base de dados. Por esta razão, é interessante criar um modelo de previsão de falhas em base de dados através da adaptação do modelo já existente na medicina o MEWS, visto que este utilizará métricas dos vários componentes da base de dados o que permitirá realizar uma análise global ao seu comportamento.

Capítulo 4

Monitorização e Prevenção de Falhas

A monitorização de falhas é um processo complexo realizado através da avaliação de dados relativos ao comportamento do sistema, nomeadamente quando este está exposto a fatores que possam originar uma falha, isto é, independentemente da área, o processo de monitorização de falhas consiste em obter informações úteis para evitar situações de futuras falhas [13].

A prevenção de falhas está muitas vezes relacionada com o conceito de risco, por isso, a sua importância é proporcional aos efeitos negativos que possam advir de uma possível falha [13]. Esta característica torna a prevenção de falhas em base de dados hospitalares uma questão essencial pois está em jogo a qualidade dos serviços prestados ao paciente.

4.1 Conceitos

Apesar dos mecanismos de tolerância a falhas implementados as **falhas** continuam a acontecer e podem ter várias origens, como por exemplo, recursos físicos que deixem de funcionar, ou erros humanos na utilização do sistema. Estas conduzem a **erros** que afetam o comportamento normal do sistema. Independentemente do sistema, quer seja computacional ou não, é caraterizado por um comportamento padrão que deve seguir, de forma a atingir os objetivos para o qual foi construído. Quando se verifica um comportamento distinto do esperado diz-se que ocorreu um **defeito** no sistema [12]. Como já foi mencionado acima, um sistema de base de dados é composto por vários componentes, por este facto, um defeito no funcionamento do sistema pode advir de uma falha de qualquer um desses componentes [12].

Por vezes, estes defeitos originam a interrupção involuntária do sistema, a esta interrupção dá-se o nome de **unplanned downtime** [11]. No entanto, pode existir necessidade de interromper o funcionamento do sistema por outros motivos, por exemplo, manutenção ou atualização de componentes. Nestes casos, é agendada uma hora mais apropriada para que se possa parar o sistema com o menor nível de prejuízo. A este tipo de interrupções do sistema dá-se o nome de **planned downtime** [11].

Como já foi mencionado no capítulo anterior, aos sistemas que continuam disponíveis mesmo na presença de falhas em componentes de hardware (discos, memória, processador) ou software (sistema operativo, SGBD, aplicações) dá-se o nome de **sistemas tolerantes a falhas** [33].

4.2 Tipos de falha

As falhas dependendo da sua origem, da base de dados e do ambiente que a rodeia, podem ser de vários tipos. Algumas falhas estão mais relacionadas com os componentes físicos (discos, memória, processador), outras relacionadas com a própria ação do utilizador. De seguida são descritos alguns tipos de falha que se podem verificar numa base de dados, neste caso particular da Oracle.

- **Falha do utilizador** - quando um utilizador, por engano, executa determinada ação que não é a correta. Por exemplo, fazer um *update* numa tabela errada, existindo assim a necessidade de fazer *rollback* [8].
- **Falha de instrução SQL** - quando existe uma falha lógica que impede a instrução de produzir o resultado esperado. Os efeitos que já tiverem sido aplicados são removidos e a base de dados volta ao estado inicial [8].
- **Falha de processo** - quando um processo termina sem que seja dada indicação para essa ação. É necessário fazer o *rollback* das operações realizadas após o último *commit*. Para além disso, é necessário libertar recursos que estejam alocados ao processo [8].
- **Falha da instância** - quando a instância deixa de estar disponível. É necessário ativar os mecanismos de recuperação de instâncias [8, 21].
- **Falha do ambiente** - quando acontece um incêndio ou existe falha de eletricidade durante bastante tempo, danificando ou impedindo o normal funcionamento da base

de dados [21].

- **Falha no disco** - quando não se consegue aceder ao disco para escrever/ler os dados. É necessário ativar os mecanismos de recuperação de disco e recuperar os ficheiros perdidos [8].
- **Falha de Memória** - quando não existe memória suficiente, a base de dados pode ter de recorrer a disco mais vezes, tornando mais lento o sistema causando a indisponibilidade do mesmo. Esta falha também pode ocorrer por falência física da memória [21].
- **Falha do Processador** - quando o processador está sobrecarregado começa a não conseguir dar resposta a todos os pedidos e pode originar o bloqueio de todo o sistema. Menos comum, mas também possível de acontecer é o processador falhar devido a problemas exclusivamente físicos [21].

Apesar da existência deste conjunto heterogéneo de falhas, esta dissertação foca-se nas falhas provenientes dos recursos físicos e da alocação dos mesmos, uma vez que estas são muito comuns e graves em bases de dados que contêm muita informação e alvo de alta utilização. É de ter em conta também que muitas vezes a indisponibilidade de uma instância está relacionada com este tipo de falhas.

4.3 Monitorização de uma base de dados

Para construir um bom sistema de prevenção de falhas é necessário antes de mais proceder à avaliação da performance dos recursos que podem vir a causar essas mesmas falhas. Para tal, é necessário recorrer ao processo de monitorização da base de dados.

A monitorização do comportamento e dos componentes de uma base de dados é vista como uma boa prática não só de análise de performance, mas também de segurança, o que leva a que as empresas apostem cada vez mais na utilização de sistemas de monitorização. A nível de segurança, o processo de monitorização consiste em identificar comportamentos e fluxos anormais na base de dados que possam indicar, por exemplo, a presença de um intruso a executar um milhão de vezes um comando, ou então, um recurso que está constantemente bloqueado por um utilizador [34].

Relativamente à monitorização como análise do desempenho de uma base de dados, pode dizer-se que é um processo de extrema importância na medida em que permite ava-

liar qual o estado atual da base de dados, identificando, caso existam, os sintomas de possíveis problemas, ou seja, promove o diagnóstico de falhas [35, 36].

São vários os parâmetros que influenciam o desempenho da base de dados, dependendo do ambiente e do vendedor da mesma. Os parâmetros mais importantes relativos aos recursos físicos são [2, 37]:

- Número de processadores - Um maior número de processadores aumenta a capacidade de processamento.
- Velocidade de acesso a disco - Diminui a perda de tempo em operações de input/output (I/O).
- Capacidade dos discos - Devem possuir uma capacidade apropriada para que não aconteçam situações de falta de espaço em disco.
- Discos em configuração RAID - Para além de melhorar a performance, promove a segurança dos dados.
- Tráfego da Rede - Pode influenciar na velocidade das iterações utilizador - base de dados.
- Capacidade de memória - Quanto maior a capacidade, mais informação pode estar em memória logo o acesso aos dados é mais rápido.

Para além dos recursos físicos acima listados é também muito importante monitorizar dados relativos ao funcionamento da base de dados, tais como informação sobre sessões e utilização de recursos a elas associada. Devido ao grande número de parâmetros e à complexidade da tarefa de monitorização, este processo pode ser dividido em três fases:

- **Definir a zona de ação** - Estabelecer quais os componentes/funções a monitorizar. Esta escolha deve ser efetuada com a ajuda dos utilizadores finais da base de dados pois são eles muitas vezes que notam as anormalidades no comportamento da base de dados [35]. Esta fase é muito importante na medida em que pretende minimizar o custo a nível de utilização de recursos, decorrente do processo de monitorização. Desta forma, diminuindo a área de ação diminui também a quantidade de dados a recolher, promovendo assim uma monitorização mais concisa e correta [10, 13].
- **Recolha das estatísticas**- Devem ser recolhidas estatísticas, ao nível do sistema computacional e ao nível do sistema Oracle relevantes para a zona de ação especificada. Estas estatísticas devem ser armazenadas para quando necessário ser possível o seu processamento [13, 35, 38].

- **Analisar estatísticas recolhidas** - É uma fase muito importante, que consiste em trabalhar as estatísticas recolhidas de forma a que elas possam transmitir informações úteis para apurar a existência de problemas de performance, ou sintomas dos mesmos, no sistema de base de dados [13, 35, 38].

4.3.1 Análise do desempenho da base de dados

A caraterização do desempenho de uma base de dados pode ser obtida através da análise das estatísticas obtidas. Existem dois métodos principais para efetuar esta análise, são eles [39]:

- **Análise baseada em rácios**- São calculados vários rácios com as estatísticas recolhidas para determinar qual o estado da base de dados. Esta técnica tem vindo a entrar em desuso, no entanto, eles são bastante úteis para os administradores das grandes bases de dados, onde é demorada a inspeção dos eventos em espera. Com a utilização de rácios é possível realizar uma rápida análise geral à base de dados, identificando as partes com má performance [39]. É importante salientar que nestes rácios, devem ser utilizadas estatísticas "delta", isto é, devem ser calculados para determinados intervalos de tempo. No caso de serem utilizadas estatísticas cumulativas podem originar uma análise incorreta [13, 39].
- **Análise baseada em bottleneck (congestionamento)** - A determinação do estado da base de dados provém da análise dos eventos que fazem interromper o normal desempenho da base de dados. É muito importante pois permite ao administrador saber onde é que se está a perder tempo na base de dados, no entanto, é uma análise mais demorada uma vez que o utilizador tem normalmente de ir descendo de camadas até chegar à sessão e muitas vezes até à instrução SQL que está a causar o problema. Este método está disponível nos mais recentes sistemas da Oracle e tem como pré-requisitos: definir o valor do parâmetro de inicialização, *timed_statistics*, para *true*; e excluir alguns eventos inativos "idle events" que não são necessários para a avaliação da performance, tais como: *lock element cleanup, pmon timer, rdbms ipc message, smon timer, SQL*Net message from client, SQL *Net break/reset to client, SQL *Net message to client, SQL*Net more data to client, dispatcher timer, Null event, parallel query dequeue wait, parallel query idle wait - Slaves, pipe get, PL/SQL lock timer, slave wait, virtual circuit status* [39, 35].

É preferível a utilização destes dois métodos em conjunto na análise da performance da base de dados, para que, um complemente o outro e seja possível efetuar uma análise completa à base de dados [39].

4.3.2 Ferramentas de monitorização

Para proceder à monitorização da base de dados pode-se recorrer a dois tipos de ferramentas: a monitorização baseada em *scripts* e através de ferramentas gráficas [40].

A monitorização baseada em *scripts* é mais adequada para bases de dados estáticas de baixo nível pois apesar de ser de fácil implementação, não é tão apelativa como as soluções gráficas. A monitorização através de *scripts* é utilizada, sobretudo, por ser mais fácil personificar os códigos de modo a se obter o resultado pretendido e também pelo facto do impacto que causa na base de dados ser muito leve. No entanto, para além destas técnicas dependerem muito do conhecimento do administrador da base de dados, são um pouco trabalhosas de implementar em base de dados complexas. A monitorização baseada em scripts pode ser implementada em sistemas Oracle através de uma das suas ferramentas de performance: as *performance views* [40].

Uma das soluções automáticas gráficas proprietárias mais conhecidas é o Automatic Database Diagnostic Monitor que realiza análises estatísticas e de desempenho em base de dados Oracle. Incluído nesta ferramenta está o Automatic Workload Repository (AWR) que é responsável por automaticamente promover a análise e processamento, por exemplo, das estatísticas das sessões/sistema e estatísticas sobre tempo de execução de instruções SQL, possibilitando assim, ao administrador do sistema ajustar o desempenho das base de dados [41].

Outra solução com alguma fama na temática da monitorização de sistemas computacionais é o software Nagios. Permite a monitorização dos vários equipamentos da rede da organização (hosts, switchs, servidores, impressoras, base de dados) e possibilita a criação de alertas, caso algum parâmetro esteja fora dos valores normais. Este software pode ser encontrado em duas versões: a versão comercial (Nagios XI) e a open-source (Nagios Core). No entanto, na versão open-source ainda não são disponibilizadas muitas ferramentas para monitorização base de dados. O programa é composto por um módulo principal aos qual podem ser adicionados plugins com o intuito de aumentar as suas funcionalidades [42]. Uma explicação mais detalhada sobre este programa pode ser encontrada

no anexo A.

Nesta dissertação, decidiu-se utilizar a monitorização baseada em scripts por várias razões, nomeadamente:

- é mais fácil definir a zona de ação;
- permite utilizar ferramentas já existentes no CHP;
- permite a criação de um programa em *background* de monitorização que pode ser facilmente integrado com a ferramenta utilizada para o tratamento e visualização dos dados, o *Pentaho Community*.

4.3.3 Ferramentas utilizadas para monitorização

O processo de recolha de estatísticas é muito importante e pode ser efetuado recorrendo a várias ferramentas. Com base nas estatísticas a recolher podem ser necessário utilizar ferramentas disponibilizadas pelo sistema de base de dados em questão, neste caso Oracle, ou então ferramentas do sistema operativo do servidor de base de dados, neste caso, sistemas Unix. Normalmente, para proceder à recolha eficaz das estatísticas chave para avaliar a performance da base de dados é necessários usar os dois tipos de ferramentas em conjunto e de forma sincronizada, permitindo posteriormente um cruzamento de dados para se obter informações ainda mais relevantes [13]. Nesta dissertação, serão utilizadas as seguintes ferramentas:

- Ferramentas do sistema de base de dados - *Performance Views*
- Ferramentas do sistema operativo - Comandos Unix

Performance Views

As várias versões de base de dados da Oracle disponibilizam ferramentas especiais que auxiliam no processo de monitorização da base de dados. Um exemplo dessas ferramentas são as *performance views (v$)*, que consistem num conjunto de objetos onde são armazenadas informações, sobre diversos aspectos da base de dados, úteis para o processo de monitorização. É importante salientar que, por razões de segurança, estes objetos apenas podem ser acedidos por utilizadores “SYS” [35, 43].

Embora o aspeto de uma *performance view* seja semelhante ao aspecto de uma tabela normal, estas não podem ser tratadas de igual forma, pois, estas *views* refletem, em tempo

real, o estado do sistema. Por esta razão, estão apenas disponíveis para consulta de informação, sendo negada qualquer outra operação. Outra característica importante é que informações constantes das *views* são atualizadas automaticamente ao longo da existência da instância da base de dados, o que leva a que estas, sejam também denominadas de **dynamic performance views** [35, 43].

Relativamente ao seu conteúdo, as *performance views* mais interessantes para o processo de avaliação de desempenho são aquelas que fornecem informação sobre estatísticas de componentes chave da base de dados, tais como, processador, memória, rede e sessões. Para monitorizar o desempenho global devem ser utilizadas *performance views* do nível de sistema. No caso de ser necessária informação mais detalhada sobre as sessões devem-se utilizar as *performance views* do nível de sessão. Podem ainda ser utilizadas vistas do nível de SQL, para identificar instruções SQL problemáticas. Na tabela 4.1, encontram-se representados alguns exemplos de *performance views* de cada um dos níveis.

Tabela 4.1: Performance Views

Estatísticas	**Nível de sessão**	**Nível de Sistema**	**Nível de SQL**
Tempo	v$sess_time_model	v$sys_time_model	
Evento em espera	v$session_event	v$system_event	
Recursos	v$sesstat, v$session, v$sessmetric	v$sysstat, v$process, v$sysmetric	v$process, v$sqlarea

De acordo com as métricas que possuem, estas *views* podem ser divididas em três grupos principais: estatísticas de tempo, estatísticas de eventos em espera e estatísticas de sessões e sistema [35, 41].

As estatísticas de tempo encontram-se nas vistas: *v$sess_time_model* (onde se registam as estatísticas temporais de cada sessão) e *v$sys_time_model* (onde se registam as estatísticas temporais do sistema). Como o próprio nome indica, estas estatísticas permitem obter informação sobre a quantidade de tempo gasto em operações efetuadas na base de dados. As estatísticas temporais mais importantes para o processo de avaliação de desempenho são [44]:

- **DB time** - Quantidade total de tempo (em microssegundos) gasto na execução de chamadas à base de dados. Bom indicador para identificar carga de trabalho, sendo normalmente proporcional à quantidade de utilizadores e de pedidos efetuados por

estes [45].

- **CPU time** - Quantidade de tempo de utilização do CPU (em microssegundos) durante a execução de chamadas à base de dados [44].
- **Background CPU time** - Quantidade de tempo de CPU consumido por processos de background [44].
- **Parse time elapsed** - Tempo despendido no *parsing* das instruções SQL [44].

As estatísticas de eventos em espera podem ser observadas nas vistas: *v$system_event* (promove uma vista geral de todos os eventos em espera no sistema da base de dados) e *v$session_event* (onde se encontram informações mais detalhadas sobre eventos em espera para cada sessão). Através destas estatísticas consegue-se obter informação sobre os eventos em espera e respetivos tempos de espera para a realização de uma tarefa. Estes tempos de espera devem-se ao facto de, por vezes, ser necessário esperar pela ocorrência de alguns eventos antes que seja possível dar continuidade ao processamento de determinada operação [41, 43]. São exemplos deste tipo de estatísticas [44]:

- **Total Waits** - Número total de esperas para que o evento ocorra [44].
- **Time Waited** - Tempo total de espera (em centésimos de segundo) pela ocorrência de um evento [44].
- **Wait class** - Nome da classe do evento, permite saber qual a origem do evento em questão [44].
- **Average Wait** - Tempo médio de espera (em centésimos de segundo) [44].

As estatísticas das sessões e sistema fornecem informações sobre a utilização de recursos do sistema. Se a informação pretendida for apenas a nível global do sistema deve-se consultar a vista *v$sysstat*, que contêm as estatísticas globais do sistema. Por outro lado, se a informação pretendida estiver no nível dos utilizadores e respetivas sessões, dependendo da estatística em questão esta informação pode ser encontrada pela junção de duas ou mais destas views: **v$session**, **v$sess_io**, **v$sesstat**, **v$sysstat** e *v$statname*. Na *v$session*, constam informações gerais sobre a sessão, nomeadamente o utilizador, a máquina, o SID (identificador de sessão). Através do SID, é então possível obter informações sobre processos de input/output acedendo à *v$sess_io*, e às demais estatísticas relacionadas, por exemplo, com a memória e o CPU através da *v$sesstat*, e *v$statname* [44]. São exemplos deste tipo de estatísticas as seguintes [44]:

- **CPU used by this session**- Tempo de utilização de CPU medido em milissegundos.

- **Session pga memory**- Memória privada utilizada (em bytes).
- **Execute count** - Número total de chamadas (de utilizador e recursivas) que executam instruções SQL.
- **User calls** - Número de chamadas de utilizador, tais como fazer login, analisar, consultar, executar.
- **User commits** - Número de *commits* feitos pelo utilizador. É utilizado frequentemente para representar o número de transações.
- **Physical reads**- Número de blocos de dados lidos do disco.
- **Consistent gets** - Número de vezes que é solicitada uma leitura do buffer da memória RAM para se obter um bloco de dados.
- **Blocks gets** - Número de vezes que é pedido um bloco que está presente na SGA.
- **Consistent changes**- Número de vezes que um utilizador faz *rollback*.
- **Blocks changes** - Número de alterações causadas aos blocos presentes na SGA por operações de update/delete.

É importante salientar que os valores destas estatísticas são cumulativos. As estatísticas relativas à *v$sesstat* são re-iniciadas quando o utilizador faz *logout*. Relativamente ao valor das estatísticas de *v$sysstat* estas são re-iniciadas quando a instância é terminada, ou seja, acumulam os valores durante a existência da instância da base de dados [35, 13].

Para além do nível de sistema e de sessões existem *performance views* que permitem obter estatísticas ao nível das instruções SQL. Por exemplo, é possível através delas identificar quais as queries SQL que demoram mais tempo. Queries SQL com elevado tempo de duração podem significar má construção da query ou que a base de dados está a demorar a responder. Por esta razão, é importante isolar a query SQL que demora mais tempo, para tal é necessário recorrer às views: *v$session* ,*v$sqlarea* ,*v$process*, das quais se podem tirar as seguintes informações:

- **Osuser** - Nome do utilizador associado à query.
- **Sql_text** - Instrução SQL.
- **Elapsed_time** - Tempo de duração da query SQL.

Para efetuar a monitorização da base de dados é aconselhado iniciar pelo nível do sistema, e só depois, se necessário aceder ao nível de sessões e por último ao nível do SQL, pois à medida que se desce de nível a complexidade das queryies SQL elaboradas para obter as estatísticas aumenta [13].

Nas versões mais atuais do Oracle, é possível ainda aceder a um conjunto de *views* que servem de base ao AWR. Nessas vistas, é possível encontrar um conjunto de métricas que auxiliam no processo de monitorização. A mais interessante é a *v$sysmetric*, onde podem ser encontradas algumas medidas relativas à performance da base de dados, como algumas já explicadas acima e com a vantagem que fornece também uma métrica relativa ao tráfego da rede onde a base de dados se encontra instalada [41].

Comandos Unix

Devido aos servidores usados no CHP serem dotados de sistemas Unix é possível recorrer a uma serie de comandos disponibilizados por estes que permitem obter informações interessantes sobre os recursos físicos do sistema computacional onde a base de dados está implementada. Os comandos que permitem obter essas informações são:

top (Top CPU Process) - Mostra e atualiza informação relativa ao top de processos no sistema. Permite também obter informação sobre a percentagem de tempo em cada um dos estados do processador [46]:

- **User**: A executar processos dos utilizadores.
- **Nice**: A executar processos prioritários.
- **System**: O tempo a executar o kernel e os seus processos.
- **Idle**: Tempo que o CPU está parado, sem processos para executar.
- **Interrupt**: Tempo que o CPU está a tratar de interrupções de hardware/software.
- **Swapper**: Espera por input/output de uma página de memória.

Com este comando, é possível ainda identificar a quantidade de memória total, utilizada e livre do sistema [46].

ps (Process Status) - Mostra os processos que estão ativos no momento, o utilizador e o tempo de atividade de cada um deles, sendo útil para verificar se determinado processo ainda está a correr [46].

sar (System Activity Reporter) - Fornece informações sobre a paginação, memória, utilização de CPU e definições do buffer, promovendo desta forma um relatório completo sobre o estado atual do sistema. Dependendo das opções especificadas no comando, este relatório pode conter a totalidade da informação ou apenas as pretendidas [46]. No caso prático deste trabalho apenas foram usadas as opções:

- **sar -r** - Obter informação sobre a utilização da memória.

- **sar -u** - Obter informação sobre a utilização do processador.

vmstat (Virtual Memory Statistics) - Disponibiliza informações sobre memória, memória virtual, utilização do processador e processos [46]. Neste trabalho, apenas foi utiliza informação sobre a memória e a utilização do processador.

4.4 Avaliação do desempenho da base de dados

4.4.1 Métricas utilizadas para avaliação do desempenho

São várias as métricas que podem ser utilizadas para avaliar o desempenho das bases de dados. Com base na literatura e após algumas experiências constatou-se que as mais apropriadas para este estudo são as métricas abaixo descritas. De acordo com a sua zona de ação estas métricas podem ser divididas em dois grupos [2, 35, 38]:

- **Métricas relacionas com os recursos físicos:**
  - **Quantidade de CPU utilizado** - O processador é um componente chave num servidor de base de dados, por isso é muito importante analisar a quantidade de CPU utilizado. Duas situações anormais podem ocorrer. Se a utilização do CPU for extremamente baixa, isto pode indicar que existe um alto nível de operações de I/O. Outro comportamento indicativo de problemas, é se o CPU estiver a ser muito utilizado mas apenas por um pequeno número de programas, isto leva a que alguns processos nunca consigam aceder ao processador. A percentagem de CPU pode ser obtida através de comandos do sistema operativo [35, 39].
  - **Quantidade de memória utilizada** - A memória é um dos componentes chave para a rapidez dos sistemas de base de dados, visto que, dependendo da localização dos dados pretendidos o tempo de resposta é influenciado. Quando os dados pretendidos se encontram em memória, as operações são efetuadas mais rapidamente. No entanto, a memória é uma estrutura física finita que não pode conter toda a informação necessária, então apenas residem lá as informações mais frequentemente pretendidas. É importante determinar quando é que existem picos de utilização de memória pois estes podem conduzir a um problema de performance do sistema, levando numa primeira fase ao aumento do tempo de resposta do sistema e se nada for efetuado pode

implicar mesmo o bloqueamento do sistema. Utilizando os comandos do sistema operativo é possível recolher a percentagem de utilização de memória [39].

- **Volume do tráfego de rede** - A rede tem bastante influência na performance da base de dados, pois todos os componentes do sistema de base de dados encontram-se ligados por rede. Para além disso, os pedidos dos utilizadores normalmente chegam através da rede. Por tudo isto, é necessário monitorizar o tráfego da rede pois um grande aumento na quantidade desta métrica pode conduzir a atrasos podendo até, num caso mais extremo, comprometer a execução dos pedidos dos utilizadores.

- **Métricas relacionas com a carga de trabalho:**

  - **Tempo de resposta** - é o tempo que decorre entre o instante da colocação da query pelo utilizador, até à receção da totalidade dos resultados, este tempo deve ser o menor possível [2, 38]. Nos sistemas Oracle, este tempo é denominado por *DB time* sendo definido como a soma do tempo total (incluí CPU time, I/O time, Wait time) gasto nos pedidos de todos os utilizadores. Por esta razão, é um bom indicador da carga de trabalho do sistema pois normalmente aumenta com o número de utilizadores e com número de pedidos, sendo que por vezes, também pode aumentar devido a de transações de maiores dimensões, ou quando existe algum problema no sistema [45, 47]. É importante salientar que tempo de CPU (CPU time) é diferente do tempo real, ou seja, num sistema de base de dados com oito processadores para cada minuto no tempo real existem oito minutos de CPU_time disponível [35].
  - **Número de transações** - é visto como a unidade de trabalho das bases de dados. Em princípio, uma base de dados tem mais ou menos trabalho de acordo com o número de transações que são realizadas. Nos sistemas Oracle o número de transações pode ser obtido através da soma dos valores das estatísticas *"user commits"* e *"user user rollbacks"* pois cada transação bem sucedida é sempre finalizada com um comando "commit" e qualquer operação desfeita é marcada como um "rollback"[2, 13, 39].
  - **Rácio de Chamadas Recursivas** - As transações são compostas por dois tipos de chamadas, as chamadas de utilizador (*"user calls"*) e as chamadas

recursivas (*"recursive calls"*). Quando, após um pedido de um utilizador, é executada apenas uma instrução SQL, é considerada uma chamada de utilizador. Se a instrução SQL solicitada pelo utilizador, necessitar de outra instrução SQL para satisfazer o pedido do mesmo, então ocorre uma chamada recursiva. Este rácio pode ser calculado através da equação [44]:

$$RC = \frac{(recursivecalls)}{(recursivecalls + usercalls)} \tag{4.4.1}$$

O ideal é que este rácio seja o mais baixo possível, pois um elevado número de chamadas recursivas pode indicar: problemas a nível do dimensionamento das tabelas; um aumento de ordenações em disco devido a problemas de memória cache e/ou excesso de *triggers* em execução [48].

- **Número de operações** - é a soma das chamadas de utilizador e chamadas recursivas. Uma transação pode dar origem a várias operações o que pode influenciar a performance do sistema, pois apesar de um número baixo de transações se destas resultarem muitas operações aumenta a carga da base de dados. É então necessário proceder à recolha dos valores da estatística *"execute count"* [13].
- **Número de sessões** - O número de sessões não está diretamente relacionado com o número de utilizadores, uma vez que, um utilizador pode ter mais de uma sessão. O número de sessões pode ser obtido através da estatística *"logons current"*. É importante determinar qual a altura do dia em que existe um maior número de sessões uma vez que a cada sessão é atribuída uma porção de memória (PGA), o que pode influenciar a performance da base de dados [35].
- **Rácio do Buffer Cache** - Este rácio representa a fração entre os acessos a disco e os acessos a memória. O seu cálculo pode ser efetuado através da equação:

$$BC = 1 - (\frac{(physicalreads)}{(consistentegets + blockgets)}) \tag{4.4.2}$$

Uma diminuição deste rácio pode ter origem numa subida do número de leituras do disco o que pode indicar problemas na memória. No entanto, é preciso ter cuidado ao efetuar este rácio, sendo necessário garantir que todas as estatísticas usadas são estatísticas "delta". Caso contrário, se forem usados

valores cumulativos, o rácio pode ser alto em virtude dos elevados valores e, no entanto, estar a passar-se algo de errado com a base de dados, não sendo visível devido à ordem de grandeza das variáveis não ser a mesma [39, 47].

- **Quantidade de pedidos de operações de I/O** - É necessário ter particular cuidado com este tipo de operações pois estas, normalmente, são mais demoradas. Um elevado número de operações de I/O pode indicar problemas na memória, existindo necessidade de ir mais vezes a disco buscar informação. O elevado número destas operações pode também estar relacionado com a escrita dos ficheiros de *redo log*. Da elevada quantidade de operações de I/O podem surgir atrasos na realização dos pedidos dos utilizadores [35].
- **Redo Size** - Representa a quantidade de entradas no ficheiro *redo log*(em bytes). É muito importante devido à função dos ficheiros *redo log* (guardar alterações efetuadas na base de dados), um aumento no valor desta estatística pode indicar um maior número de operações sobre os dados e consequentemente uma maior carga de trabalho para a base de dados [44].
  b
- **Pedidos de espaço para buffer redo** - Quando não existe espaço suficiente no *redo log uffer* é necessário pedir mais espaço para não comprometer o registo das operações efetuadas na base de dados. O espaço é obtido através da escrita das informações do *buffer* para ficheiro o que pode levar a atrasos. Isto pode acontecer devido a um aumento de transações, espaço insuficiente reservado para o *redo log buffer* ou ainda que a periodicidade de escrita dos dados do buffer para o ficheiro não é a mais adequada [48].

Relativamente à **existência de eventos em espera**, é muito importante proceder à sua análise para identificar quais as causas da perda de tempo na base de dados. Para tal, numa primeira fase é necessário excluir os eventos pertencentes à classe “idle”, ou seja, aqueles que estão relacionados com processos que apenas são executados em intervalos de tempo pré-definidos cujo o tempo de espera é normalmente elevado, no entanto, sem influência na performance do sistema. De seguida, deve-se ordenar os restantes eventos por o tempo total de espera formando assim o *top* de eventos em espera que é um bom indicador de performance. Através do evento, pode-se chegar às causas que o originam, na Tabela 4.2 encontram-se alguns dos eventos mais importantes e as suas possíveis causas [35, 41].

Tabela 4.2: Principais "Wait Events" e possíveis causas

Wait Event	**Definição**	**Possível Causa**
buffer busy waits	Espera para aceder a um bloco do buffer da cache.	Existir vários processos a aceder concorrentemente ao mesmo bloco
free buffer waits	Não existem buffers livres, então é necessário escrever um buffer utilizado na memória para disco.	Processo DBWR lento, Sistema de I/O muito lento
db file scattered read	Um processo está a ler buffers não contínuos da SGA e está à espera de uma chamada I/O para retornar	Problema na query SQL, Sistema de I/O lento
db file sequential read	Um processo está a ler um bloco da SGA e está à espera de uma chamada I/O para retornar	Problema na query SQL, Sistema de I/O muito lento
log buffer space	Quando processo está à espera de um lugar livre no buffer de log	Log Buffer pequeno, Sistema I/O lento
log file sync	Espera por escrita no ficheiro redo log antes de concluir o commit ou rollback de uma operação.	Discos Lentos
log file parallel write	Escrita de registos de redo para os ficheiros de redo log do buffer de log.	Discos Lentos

É importante salientar que todas estas métricas não são estáticas, ou seja, os valores resultantes podem variar de acordo com o ambiente e com o estado do ambiente em questão. Por exemplo, o facto de o número de sessões não ser fixo, pode levar a desempenhos diferentes da base de dados em instantes temporais distintos. Note-se que, num dia com mais utilizadores a aceder à base de dados, a mesma query pode demorar mais tempo a ser resolvida do que num dia em que o número de utilizadores seja reduzido. Por conseguinte, é necessário monitorizar frequentemente o desempenho da base de dados [37].

4.4.2 Representação do comportamento padrão das bases de dados

Com base nas métricas recolhidas é possível representar um padrão de utilização da base de dados. Para tal, foram usadas as seguintes medidas:

- **Média** - Pode ser definida como a fração entre o somatório dos valores recolhidos e o número de recolhas. Esta medida é bastante influenciada por valores que se encontrem muito afastados da média [49]. Em SQL, pode ser facilmente calculada recorrendo à função *avg*.
- **Desvio Padrão** - É utilizado para medir a variabilidade dos dados em relação à média. Pode ser definido como a raiz quadrada da variância (média aritmética dos desvios em relação à média aritmética) [49]. Em SQL, pode ser calculado facilmente através da função *stddev*. De seguida, apresenta-se o código 4.1 que permite calcular a média e o desvio padrão, de cada dia, para a métrica "utilização de CPU":

Código 4.1: Query SQL usada para o cálculo da média e desvio padrão

```
1 select to_char(datasistema,'MM/DD') as data,
2 avg(value) as média, stddev(value) as desvp
3 from r_registo_sonho where name='cpu'
4 group by to_char(datasistema,'MM/DD')
```

- **Percentis** - Dividem a serie de dados recolhidos em 100 partes iguais. O valor situado no percentil n indica que n% dos dados são menores que ele. Os percentis podem também ser utilizados em conjunto, por exemplo, utilizando o percentil 25 e o percentil 75 é possivel definir um intervalo no qual estão contidos 50% da totalidade de dados recolhidos [49]. É importante salientar que os dados devem-se encontrar ordenados para efetuar o cálculo dos percentis. Neste trabalho, os

percentis foram calculados com o auxílio da função "percentile_cont". De seguida, encontra-se o código 4.2 usado para o cálculo dos percentis:

Código 4.2: Query SQL usada para o cálculo dos percentis

```
1  select datasistema, name,
2  percentile_cont(0.25)
3  within group (order by value asc) p25,
4  percentile_cont(0.75)
5  within group (order by value asc) p75,
6  percentile_cont(0.80)
7  within group (order by value asc) p80,
8  percentile_cont(0.90)
9  within group (order by value asc) p90
10 from r_registo_aida
11 group by datasistema, name ;
```

4.4.3 Visualização dos resultados

O processo de monitorização, mesmo após a definição da zona de ação, é um processo do qual resultam muitos dados de difícil interpretação. Para a manipulação, interpretação e visualização desses dados utilizou-se uma ferramenta de Business Intelligence (BI).

O termo BI foi introduzido por Howard Dresner do grupo Gartner, no ano de 1989, com o intuito de descrever um conjunto de conceitos, métodos, processos e ferramentas que através da utilização das novas tecnologias permitiam melhorar o negócio auxiliando na tomada de decisão [50]. Este objetivo é conseguido, através de várias etapas. É necessário em primeiro lugar proceder à obtenção e integração dos dados das diferentes fontes da empresa, armazenando-os numa grande base de dados (data warehouse). Sendo depois possível efetuar várias análises de onde se obtêm informações úteis para a tomada de decisões e experiências passadas que permitem desenvolver uma excelente compreensão da dinâmica empresarial [50, 51, 52]. Algumas das ferramentas utilizadas para a realização destas análises são: produção de relatórios e queries efetuados ao utilizador final, OLAP (Online Analytical Processing), *dashboards*, *data mining*, e ferramentas de planeamento e de modelação. A utilização de BI nas organizações traz muitos benefícios, dos quais se salientam os seguintes: economia de custos na consolidação dos dados, economia de tempo para os utilizadores dos sistemas, mais e melhor informação, melhores decisões e

mais apoio para a realização de objetivos/negócios [51].

Após a leitura de alguns estudos sobre ferramentas *open-source* de BI, constatou-se que a versão *Comunnity* do *Pentaho*, era a mais indicada para os objetivos propostos [53, 54]. A ferramenta *Pentaho* foi lançada em 2004 possuindo uma versão paga (*Enterprise*) e uma versão gratuita denominada por *Community*. A versão gratuita possui, entre outras, as seguintes funcionalidades: criação de relatórios, gráficos e *dashboards*; aplicação de técnicas de *data mining* e exportação de dados. O *Pentaho Community* é constituído por várias aplicações todas elas escritas em *Java*, o que permite que seja utilizado em todas as plataformas computacionais. Contém um módulo principal, o *bi-platform* ao qual podem ser adicionados módulos extra responsáveis por disponibilizar outras funcionalidades [53, 55]:

- **Pentaho Data Integration (Spoon)** - Permite a extração, integração e transformação dos dados. Muito útil para as tarefas de integração de dados de diferentes fontes e para a aplicação de operações matemáticas sobre os dados. Para além disso, pode ser utilizado para simplificar queries SQL, uma vez que contém um conjunto de passos (*steps*) que automatizam tarefas que seriam complexas de executar recorrendo apenas ao SQL [55].
- **Mondrian** - Permite o processamento de informação histórica armazenada através de técnicas de OLAP, ou seja, é possível proceder à análise dos dados sobre várias perspetivas organizando a informação em estruturas denominadas de cubos [55].
- **Pentaho Report Designer** - Permite a criação de relatórios de uma forma fácil e intuitiva. É mais completa do que a ferramenta de criação de relatórios existente no *bi-platform*. Com este módulo é possível integrar gráficos dentro dos relatórios bem como a criação de relatórios dinâmicos [55].
- **Community Dashboard Editor (CDE)** - Permite desenvolver *dashboards* que mostram de uma forma mais amigável e completa os dados recolhidos/tratados. Este módulo foi desenvolvido e é mantido pela Webdetails. O *front-end* é baseado em HTML, sendo que os gráficos e as tabelas podem ser populados com dados oriundos de varias fontes, nomeadamente: Queries SQL, ficheiros XML, cubos do Mondrian e transformações elaboradas no Spoon [55, 56].
- **Weka** - Utilizado para a aplicação de técnicas de *data mining* aos dados. Note-se que o Weka não é um módulo do Pentaho mas sim uma aplicação, também ela

gratuita, que pode ser interligada com o *Pentaho*, através do plugin Weka Scoring Plugin, no módulo Spoon [55].

No anexo B encontra-se um manual de instalação e utilização do *Pentaho Community*.

Capítulo 5

Caso de estudo

Neste capítulo procede-se à descrição do caso de estudo contextualizando o mesmo através da descrição dos sistemas de informação hospitalar e da apresentação das bases de dados para as quais se pretende desenvolver um modelo de monitorização e prevenção de falhas. Ainda neste capítulo são apresentados os passos seguidos para a concretização deste objetivo.

5.1 Evolução dos sistemas hospitalares

As instituições hospitalares, ao longo dos anos, têm vindo a atualizar e a proceder à informatização de muitos dos seus serviços, para trazer à área da saúde novas metodologias de resolução de problemas que possam melhorar constantemente a qualidade dos serviços de saúde [57, 58].

Neste sentido, surgem os sistemas de informação hospitalar. Estes sistemas são responsáveis por otimizar todo o conjunto de informação gerado nas unidades hospitalares. Para tal, devem proceder à recolha, ao armazenamento, ao tratamento e gestão de dados de todos os intervenientes (pacientes, médicos, enfermeiros, etc.) e de todos os serviços (administrativos, clínicos, etc.). Desta forma, numa determinada situação, os sistemas de informação hospitalar possibilitam, ao pessoal autorizado, um acesso rápido e eficaz à informação relevante para esta. Rapidamente estes sistemas assumiram um papel fundamental nas instituições hospitalares tornando-se ferramentas importantes nas diversas áreas da instituição, desde a gestão administrativa à prestação de atos clínicos [57].

Na década de 80, a pedido do Ministério da Saúde Português, foi desenvolvido pelo

Instituto de Gestão Informática e Financeira da Saúde (IGIF) o sistema integrado de informação hospitalar (SONHO). Este sistema tem como objetivo juntar num só sistema dados administrativos e clínicos para que possa ser consultada, em tempo real, toda a informação útil para o tratamento do paciente. No CHP, esta informação encontra-se armazenada numa base de dados denominada de SONHO [59].

A evolução destes sistemas foi a génese do processo clinico eletrónico (PCE), pois tornou imprescindível os registos clínicos em formato eletrónico. O PCE é um conjunto de documentos eletrónicos que contêm informações (clinicas, administrativas, financeiras) relativas ao paciente que possibilitam uma serie de funcionalidades como é o caso do auxílio na tomada de decisão. Devido à natureza pessoal das informações contidas no PCE, os documentos eletrónicos têm de ter em conta não só a legislação e as regras éticas, mas também a qualidade da informação [60, 61, 62]. Para além, deste sistema surgiram dois novos dois novos módulos [59]:

- Sistema de apoio médico (SAM) - Apenas pode ser acedido por médicos e contém várias opções desde registos de consulta até prescrições de meios complementares de diagnóstico [59].
- Sistema de apoio à enfermagem (SAPE) - Utilizado pelos enfermeiros para registar e consultar os cuidados de enfermagem recebidos pelo paciente [59].

No entanto, devido à complexidade de uma unidade hospitalar para além dos sistemas apresentados acima existem outros sistemas que geram distintas fontes de informação. Esta heterogeneidade tornava difícil a comunicação entre os vários sistemas, e por conseguinte impedia a interoperabilidade [60, 63].

A solução passou por desenvolver, uma ferramenta dinâmica: a AIDA (Agência para a Integração, Difusão e Armazenamento da informação) que procede à partilha da informação e do conhecimento entre os vários sistemas de informação e equipamentos hospitalares [60, 63]. A AIDA foi desenvolvida por um grupo de investigadores do Departamento de Informática e do CCTC da Universidade do Minho. Esta agência é constituída por agentes responsáveis por promover a comunicação através do envio/receção de informação, e por gerir e armazenar essa informação. Deste modo, a AIDA promove a interligação e a partilha de informação entre os vários equipamentos e sistemas de informação, nomeadamente [60, 61]:

- Sistema de informação administrativo - Responsável por gerir e arquivar as infor-

mações administrativas relativas ao tratamento do doente [60, 61].

- Sistema de suporte médico - Responsável por gerir e arquivar a informação clínica [60, 61].
- Sistema de PCE - Disponibiliza um conjunto de registos médicos dos vários serviços, patologias e pacientes [60, 61].
- Sistema de informação de enfermagem - Responsável por gerir e armazenar informações dos cuidados de enfermagem prestados ao paciente [60, 61].
- Sistemas de informação de cada departamento - Responsáveis por gerir e armazenar as informações relativas aos serviços prestados em cada departamento. Um dos departamentos mais críticos é o de radiologia uma vez que é necessário lidar com muitas imagens no formato DICOM, aumentando a quantidade de informação a ser trabalhada [60, 61].

Através do uso da AIDA é possível garantir que os diferentes sistemas comuniquem e cooperem, uns com os outros, com vista a melhorar os serviços da unidade hospitalar. Para tal, toda a informação hospitalar importante para o processo de interoperabilidade é armazenada numa grande base de dados que tem de estar permanentemente disponível, uma vez que, contém informações vitais para a resolução dos problemas dos pacientes e para a gestão hospitalar [11].

Neste sentido, pretende-se desenvolver um modelo de monitorização e prevenção de falhas que possa ser aplicado às bases de dados SONHO e AIDA do CHP.

5.2 Base de dados SONHO

A base de dados SONHO armazena os dados provenientes dos módulos: SONHO, SAPE, SAM. Existem dois servidores que dão acesso a uma base de dados partilhada, formando a arquitetura física da Figura 5.1. O servidor **chpsonho** comporta um nodo lógico dotado de um sistema Oracle 7.3, relacionado com o serviço SONHO, e o servidor **hsa-sonho** comporta nodos lógicos, relacionados com a parte administrativa, urgências e maternidade, dotados de sistemas Oracle 9.2. Nesta dissertação apenas será abordado com maior cuidado o servidor relativo ao serviço SONHO.

Nesta arquitetura, os recursos físicos principais com influência na performance da base de dados são: os conectores de alta velocidade, os switchs, os servidores, e a confi-

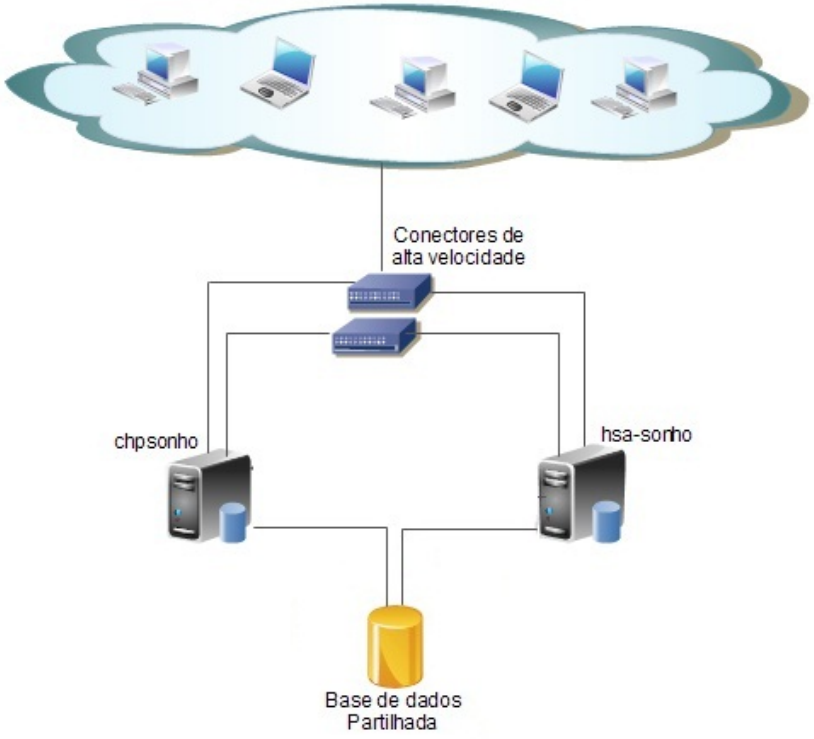


Figura 5.1: Arquitetura da base de dados SONHO do CHP

guração dos discos utilizados na base de dados partilhada.

Tanto os conectores de alta velocidade como os *switchs* funcionam a Gigabyte, o que permite um rápido acesso por parte dos servidores à base de dados partilhada.

Os componentes mais complexos são os servidores, uma vez que, possuem vários itens que influenciam o desempenho do sistema. Na Tabela 5.1, encontram-se representadas as especificações relativas ao processador e à memória do servidor de acesso ao serviço SONHO.

Em relação à área de armazenamento de dados esta é composta por uma série de discos em configuração RAID, promovendo desta forma uma maior disponibilidade dos dados.

É importante salientar que alguns dos dados armazenados na base de dados SONHO, nomeadamente, os importantes para o processo de interoperabilidade são também armazenados na base de dados AIDA.

Tabela 5.1: Especificações dos componentes dos servidores do SONHO

Item	**chpsonho**
CPU	PA 8900 CPU Module 3.2
CPU Mhz	999
Nº CPUs	8
Cores por CPU	1
Total de Memória Física	16397 MB
Memória física disponível	990 MB
Swap Max.	16384 MB

5.3 Base de dados AIDA

A base de dados que serve de suporte à AIDA do CHP esta é dotada de um sistema Oracle RAC 11G. Este sistema comporta dois nodos, sendo que em cada um deles encontra-se instalada uma instância da base de dados e uma instância do ASM.

Juntamente com este sistema, existe mais um nodo que faz a ligação com uma base de dados *standby*, implementando desta forma um sistema Oracle Data Guard. Como já foi mencionado acima, esta arquitetura permite em caso de falha dos nodos principais, ou da base de dados primária, aceder aos dados através da base de dados *standby*. É importante que a replicação seja feita periodicamente, permitindo assim a diminuição do desfasamento dos dados nas duas bases de dados. Esta arquitetura pode ser facilmente visualizada na Figura 5.2.

Os componentes físicos importantes para o desempenho do sistema são os mesmos que foram mencionados acima para a base de dados SONHO. As características de cada um deles não variam muito, sendo que os conectores e switchs utilizados rondam também os Gigabyte de velocidade o que um rápido acesso aos dados e uma rápida comunicação entre nodos. Relativamente aos discos, estes apresentam também uma configuração RAID promovendo a eficaz replicação dos dados.

Na Tabela 5.2, encontram-se representadas as especificações relativas a processador e à memória de ambos os servidores da base de dados AIDA.

A utilização desta configuração permite ao hospital estar bem preparado no que diz respeito à tolerância de falhas, no entanto, não possui qualquer mecanismo que se debruce

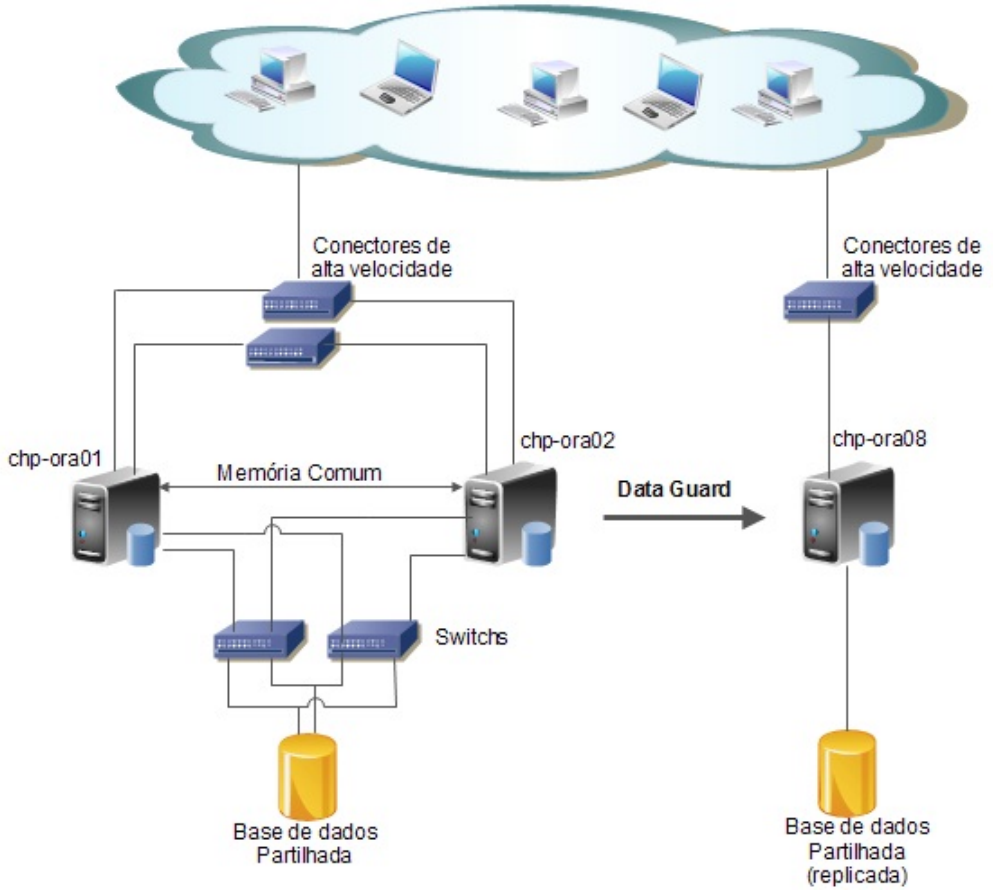


Figura 5.2: Arquitetura da base de dados AIDA do CHP

Tabela 5.2: Especificações dos componentes dos servidores

Item	**CHP-ORA01**	**CHP-ORA02**
CPU	Quad-Core AMD Opteron(tm) Processor 8354	Quad-Core AMD Opteron(tm) Processor 8354
CPU Mhz	2210.187	2210.187
Nº CPUs	4	4
Cores por CPU	4	4
Total de Memória Física	33779 MB	33779 MB
Memória física disponível	813 MB	1775 MB
Swap Max.	51609 MB	51609 MB

sobre a temática da previsão de falhas.

5.4 Metodologia

A primeira fase desta dissertação focou-se em dois aspectos fundamentais. Por um lado, procedeu-se à recolha de material bibliográfico relacionado com bases de dados, sistemas de informação hospitalar, monitorização de performance e prevenção de falhas em bases de dados. Por outro lado, foi realizado um conjunto de experiências para identificar que ferramentas utilizar, que métricas recolher e como proceder a essa recolha. Relativamente às ferramentas a utilizar a escolha recaiu sobre as *performance views* disponibilizadas pela Oracle, sobre os comandos *sar, vmstat* do sistema operativo, e sobre a ferramenta *Pentaho Community*. De forma a limitar a quantidade de dados recolhida, com base na literatura, foram consideradas apenas algumas métricas relacionadas com a performance da base de dados e com maior impacto para a previsão de falhas, nomeadamente: Número de sessões, utilização do processador, utilização de memória, número de transações, número de operações, número de operações I/O, tempo de resposta, tamanho do ficheiro *redo log*, rácio do buffer da cache, rácio de chamadas recursivas, volume do trafego da rede e número de pedidos de espaço para o buffer redo.

A segunda fase consistiu na elaboração de um programa de monitorização de falhas. Este programa de monitorização foi desenvolvido em *Java* e o seu objetivo era proceder à recolha de estatísticas relativas à utilização de recursos e ao normal funcionamento das bases de dados SONHO e AIDA, do CHP, de forma a caraterizar a carga de trabalho das mesmas. O programa é composto por vários procedimentos principais e pequenas funções que são utilizadas pelos procedimentos. Um dos primeiros procedimentos a salientar é o **conexaosrv()**, sendo que este procedimento permite uma ligação remota à base de dados que se deseja monitorizar. Este acesso tem de ser efetuado com uma conta que possua privilégios de acesso às *performance views* para que seja possível executar a query presente no código 5.1.

Código 5.1: Query SQL para recolha das métricas das performance views

```
1 select to_char(trunc(sysdate,'hh24') + (trunc(to_char(sysdate,'mi')/15)
    *15)/24/60,'hh24:mi') grpdate, name, value
2 from v$stat_name, v$sysstat where name in ('user commits','user
    rollbacks',
```

```
3  'execute count','user calls','recursive calls',
4  'logons current','consistent gets','physical reads','db block gets',
5  'redo size','redo log space requests',
6  'physical read total IO requests','physical write total IO requests')
7  union all
8  select TO_CHAR(trunc(sysdate,'hh24') + (trunc(to_char(sysdate,'mi')/15)
       *15)/24/60,'hh24:mi') grpdate,
9  stat_name name, round(value*0.000001,0) from v$sys_time_model
10 where stat_name ='DB time'
11 union all
12 select TO_CHAR(trunc(sysdate,'hh24') + (trunc(to_char(sysdate,'mi')/15)
       *15)/24/60,'hh24:mi') grpdate,
13 metric_name as name, round(value,2)
14 from v$sysmetric
15 where metric_name='Network Traffic Volume Per Sec'
16 order by name
```

Com o código 5.1, é então possível aceder às três *performance views* que contêm o valor das métricas utilizadas. É importante salientar que esta query apenas é válida para as duas instâncias da AIDA. Na versão Oracle 7 do SONHO as *views vsys_time_model, vsysmetric* ainda não se encontram implementadas, ou seja, na versão do programa para a base de dados SONHO é utilizada apenas a primeira parte da query. O **conexaocli()** é um procedimento análogo ao primeiro, no entanto, o seu objetivo é a ligação à base de dados na qual serão armazenados os dados provenientes da monitorização.

Como para a recolha de algumas métricas é necessário a execução de comandos do sistema operativo foi necessário criar uma nova classe em *Java* para estabelecer uma ligação por SSH (Secure Shell) à linha de comandos do sistema operativo. Desta foma é possível a execução dos *scripts* responsáveis por recolher a utilização do processador e da memória. O código 5.2, é um exemplo desses *scripts*.

Código 5.2: Script bash para recolha da utilização do processador

```
1 #!/bin/bash
2 sar -u 2 5 > auxcpu.txt
3 cat auxcpu.txt | tail -n1 |cut -c 27-36
```

Este script procede à recolha dos valores de utilização de processador durante cinco intervalos de dois segundos, armazenando as medições num ficheiro auxiliar. Após este

período de tempo é extraído o campo correspondente à média das medições de percentagens de utilização do processador por parte dos processos dos utilizadores. Os outros *scripts* e respetiva explicação podem ser encontrados no anexo C.

Outro procedimento importante é o **diferenca(ResultSet in, ResultSet fi)**. Como já foi visto anteriormente os valores armazenados nas *performance views* são cumulativos. Através deste procedimento é possível proceder ao cálculo por intervalos, neste caso intervalos de um minuto. Neste procedimento são também calculados os rácios e as métricas que não podem ser retiradas automaticamente das *performance views* como é o caso do número de transações. O procedimento recebe como parâmetros o resultado da execução do código 5.1 no instante x e no instante x mais um minuto.

O programa de monitorização encontrou-se em execução aproximadamente durante um mês e meio, sendo que os valores obtidos foram armazenados num conjunto de tabelas (anexo D). Com base nestes dados construiu-se a curva de valores padrão ao longo do dia para cada métrica. Com vista a determinar quais os limites de normalidade foram utilizados percentis. Sendo assim, para limite superior foi escolhido o percentil 75 e para limite inferior o percentil 25, garantindo desta forma que 50% dos dados se encontram dentro destes limites.

Após a determinação dos limites foi então possível, utilizando a ferramenta *Pentaho Community*, elaborar alguns *dashboards* contendo tabelas e gráficos representativos do normal comportamento das bases de dados ao longo do dia. No caso da base de dados SONHO, foram elaborados os seguintes *dashboards*:

- **sonho_picos.wcdf** - Este **dashboard** é composto por um gráfico circular e um tabela. No gráfico estão representadas as horas onde ocorrem os valores mais elevados para cada estatística. Ao carregar na respetiva fatia, automaticamente é apresentada a descrição detalhada desses picos na tabela. Permite identificar qual o período mais crítico do dia, no qual se deve estar com particular atenção.
- **sonho_workload.wcdf** - Este *dashboard* apresenta os gráficos com os limites de normalidade para cada métrica, necessitando apenas de se efetuar a escolha da métrica na lista apresentada. Como o gráfico está apenas de hora a hora, ao clicar na hora é possível obter a variação do valor da métrica escolhida ao longo dessa hora no segundo gráfico. Contém também uma tabela, onde se encontra calculada a média e o desvio padrão para cada métrica avaliada.

O sonho_workload.wcdf tem por base várias queries, sendo um exemplo o código 5.3

.

Código 5.3: Exemplo de query sonho_workload.wcdf

```
1 select datasistema, p25, p75 from normal_stat_sonho
2 where datasistema like '%:00'and
3 name=(select cod from nomes where descri=${metrica})
4 order by datasistema
```

Esta query pretende mostrar o valor dos limites em cada hora ao longo do dia para a métrica selecionada. A métrica é selecionada numa caixa de seleção e depois passada para a query como parâmetro, denominado neste caso de "metrica". O mesmo raciocínio é usado para o *dashboard* da base de dados AIDA.

Para a base de dados AIDA, foram elaborados também dois *dashboards*:

- **aida_picos.wcdf** - Este *dashboard* é similar ao do SONHO, no entanto, apresenta dois gráficos e duas tabelas. Desta forma é possível encontrar informação sobre os dois nodos da base de dados AIDA.
- **aida_workload.wcdf** - Este **dashboard** é similar ao seu homónimo do SONHO. Com a particularidade que está associado aos dois nodos da AIDA. É possível obter o comportamento normal do chp-ora01 (nodo 1) e do chp-ora02 (nodo 2) através dos quatro gráficos para esse efeito. Para além disso, apresenta uma tabela onde é calculada a média das métricas, a carga total média da base de dados e ainda a variação de carga entre os dois nodos.

O *dashboard* **aida_workload.wcdf** é também ele suportado por uma serie de queries. Um exemplo dessas queries é o código 5.4.

Código 5.4: Exemplo de query aida_picos.wcdf

```
1 select k, descri, valor from
2 (
3 select substr(t.datasistema,1,instr(t.datasistema,':')-1) || ':00' h,
4 t.name, round(t.p75,2) valor from(select name as d, max(p75) as Mx
5 from normal_stat_aida
6 group by name) x join normal_stat_aida t on  x.Mx =t.p75
7 ) t1, nomes n where h=${hora} and name !='bc' and t1.name = n.cod
```

Com esta query pretende-se obter os máximos registados na hora selecionada. Para além disso, pretende-se obter informação sobre as métricas associadas a esses máximos, o que

torna a query um pouco mais complexa. O mesmo raciocínio foi usado par o *dashboard* **sonho_workload.wcdf**

Na segunda fase, foi então construída uma tabela de decisão (ver Tabela 5.3). Foi associado um score, em função do desvio dos valores normais. Esse desvio foi representado utilizando os percentis.

Tabela 5.3: Scores indicadores de gravidade

SCORE	**0**	**1**	**2**	**3**
Valor medido	$< p75$	$> p75 < p80$	$> p80 < p90$	$> p90$
Gravidade	Normal	Pouco Grave	Grave	Muito Grave

Procedeu-se então à alteração do programa de monitorização para que este detetasse situações anormais. Foram então adicionados mais três procedimentos: **score(String hora)**, **sendEmail(String msg)**, **detecta_falha(String causa), actualiza_limites**.

O **score(String hora)** é responsável pela implementação da tabela 5.3 em código *Java*. Em cada recolha o valor obtido para cada estatística é comparado com os limites definidos, sendo o valor do score incrementado de acordo com a situação em que o valor recolhido se encontra. É importante salientar que para as métrica "buffer cache" (no caso da AIDA e SONHO) e memória (no caso do SONHO), os scores foram associados por ordem inversa, isto é, o score 3 ficou associado a valores abaixo do percentil 75, pois como já foi visto anteriormente, nestes casos, são estes os valores indicativos de problemas. A cada intervalo de quinze minutos, é calculado o score médio das medições efetuadas nesse intervalo e se este for superior a quatro o procedimento **sendEmail(String msg)** é invocado para enviar um email com o registo das métricas e dos valores que originaram este aviso.

Dependendo do score, em cada intervalo, podem acontecer três tipos de situações:

- **pouco graves** - quando o valor medido para uma determinada estatística é superior ao valor do percentil 75. Nesta situação é emitido um sinal visual no *dashboard* e o score atribuído à métrica em questão é 1.
- **situações graves** - quando o valor medido é superior ao percentil 80 e inferior ao percentil 90, novamente é mostrado um aviso no *dashboard*, no entanto, o score associado é de 2.
- **Muito Grave** - quando o valor medido de uma estatística está fora do percentil 90

(score 3) e/ou então a soma dos scores das outras métricas é maior que 4. Nestas situações é enviado um e-mail para o administrador da base de dados alertando-o de uma possível falha.

Este programa é atualizável, pois novos limites são calculados no final do dia, se entretanto não ocorrer nenhuma falha. Esta operação é efetuada através do procedimento **actualiza_limites()**. Esta é uma situação frequente, uma vez que, a base de dados pode apresentar picos de carga não esperados e continuar operacional. Para além disso, o procedimento **detecta_falha(String causa)** regista as causas de interrupção do programa, útil para se saber quais as causas de uma possível falha.

Para representar graficamente os resultados da nova versão do programa foi então desenvolvido mais um conjunto de *dashboards*. Três deles relativos ao processo de monitorização e indicação visual de alertas (**sonho_det.wcdf, aida01_det.wcdf, aida02_det.wcdf**). Nestes *dashboards* encontram-se os gráficos relativos a cada métrica onde constam os limites (p75,p80,p90) e os valores medidos possibilitando a observação da variação de cada métrica. É importante salientar que estes gráficos são atualizados dinamicamente e mostram a evolução das medições durante a meia hora anterior.

Na base destes gráficos estão queries do mesmo género da apresentada no código 5.5.

Código 5.5: Exemplo de query dos gráficos dinâmicos

```
1 select to_char(r.datasistema,'hh24:mi'), p75, p80, p90, value
2 from normal_stat_aida n, r_registo_aida r
3 where n.name='mem' and n.name=r.name and n.datasistema BETWEEN
4 to_char(sysdate-0.5/24,'hh24:mi') and to_char(sysdate,'hh24:mi')
5 and TO_CHAR(r.datasistema,'DD/MM') = to_char(sysdate,'DD/MM')
6 and TO_CHAR(trunc(r.datasistema,'hh24') +
7 (trunc(to_char(r.datasistema,'mi')/15)*15)/24/60,'hh24:mi')=
8 n.datasistema order by r.datasistema
```

Dependendo da métrica ou da base de dados é necessário apenas adaptar a query mudando o nome das tabelas ou da métrica a usar.

Na tentativa de minimizar o número de paneis a visualizar foram elaborados o **sonho_geral.wcdf** e o **aida_geral.wcdf**. Estes *dashboards* apresentam uma tabela com a variação do valor da métrica, ou seja, indica se a métrica está ou não acima do limite. Após carregar no nome da métrica é apresentado o gráfico detalhado. A vantagem deste quadro geral é mais evidente no caso da AIDA pois possibilita a observação do estado dos

dois nodos em apenas um *dashboard*.

Por último, foi elaborado um *dashboard* (**anormal.wcdf**) que apresenta as principais conclusões retiradas através das anormalidades detetadas. No anexo E, encontram-se alguns excertos dos *dashboards* mencionados.

Capítulo 6

Apresentação e discussão dos resultados

Neste capítulo serão apresentados e discutidos os resultados provenientes da execução deste estudo. Como em cada uma das fases desta dissertação são gerados vários resultados, decidiu-se separar a sua apresentação e respetiva análise em duas partes:

- Análise dos resultados relativos ao comportamento padrão das bases de dados - caraterização da carga de trabalho (workload) das bases de dados AIDA e SONHO.
- Análise dos desvios obtidos em relação à normalidade - implementação de um modelo de monitorização e prevenção de falhas (adaptação do MEWS).

6.1 Comportamento padrão das bases de dados

6.1.1 Base de dados AIDA

Como já foi mencionado anteriormente a arquitetura da base de dados AIDA do CHP é composta por dois servidores, onde se encontram instaladas as instâncias que acedem ao mesmo repositório físico de dados. Na Tabela 6.1 encontram-se várias informações sobre cada nodo individual e também alguns resultados gerais úteis para o processo de caraterização de carga. Foi calculada a média de cada métrica para o chp-ora01 (nodo 1) e para o chp-ora02 (nodo 2), o que permitiu o cálculo do total médio para cada métrica. Através da coluna "Total", da Tabela 6.1, é possível verificar que a base de dados AIDA possui uma elevada utilização, apresentando um número médio de sessões na ordem das 681 sessões. Para além disso, é possível observar que em média são executadas cerca de 214 transações por segundo, resultando destas cerca de 742 operações por segundo na

Tabela 6.1: Resumo métricas relativas à carga de trabalho da AIDA

Nome	Média (nodo1)	DesvPad (nodo1)	Média (nodo2)	DesvPad (nodo2)	Total (nodo1+nodo2)	Comparação (nodo1-nodo2)
Nº de transações por segundo	111.74	57.086	103.236	53.622	214.976	
Percentagem de utilização do processador	15.635	7.964	17.579	8.119		
Percentagem de utilização da memória	97.624	1.682	97.466	2.464		
DB time por segundo	3.412	1.891	3.089	1.668	6.501	
Nº de sessões	346.478	84.388	334.724	82.425	681.202	
Nº de pedidos I/O por segundo	330.199	262.311	302.082	253.43	632.281	
Nº de operações por segundo	382.719	219.99	360.228	215.814	742.947	
Rácio Buffer Cache	0.998	0.008	0.992	0.114		
Quantidade de entradas redo (kb/s)	71.334	253.316	81.283	319.873	152.617	
Rácio Chamadas Recursivas	0.14	0.105	0.138	0.104		
Pedidos de espaço para redo log por segundo	0.544	2.496	0.005	0.115	0.549	
Volume tráfego de rede (bytes/s)	355063.636	149623.925	331071.917	138910.723	686135.553	

Showing 1 to 12 of 12 entries

base de dados o que comprova que esta é uma base de dados com uma carga de trabalho bastante elevada. A alta percentagem de utilização de memória é outro dos aspectos a ter em conta, uma vez que esta está diretamente relacionada com a rapidez de resposta da base de dados, e os valores acima dos 95% podem já ser considereados preocupantes. Para as métricas: percentagem de uso de processador e de memória, e para os rácios buffer cache e chamadas recursivas não foi calculado o total uma vez que estas dizem respeito a cada servidor em particular, não sendo possível utilizar o mesmo raciocínio das restantes métricas.

Apesar destes valores médios bastante elevados, há que ter em conta também os valores elevados dos desvios padrões. Nos dois nodos e em quase todas as métricas os desvios padrões são valores relativamente altos. Estes são explicados através da variabilidade dos dados recolhidos. Isto acontece sobretudo por duas razões, a primeira é que os valores para algumas das métricas variam substancialmente ao longo do dia, de acordo com a maior ou menor utilização da base de dados. A outra razão é que devido à função central que a AIDA desempenha no CHP, frequentemente são adicionadas novas aplicações e funcionalidades que resultam num aumento de carga da base de dados ao longo dos dias. Este último motivo é o que tem maior impacto na variabilidade dos dados recolhidos.

Ainda de acordo com a Tabela 6.1, é possível verificar que o nodo 1 apresenta valores

médios mais elevados do que o nodo 2 em todas as métricas exceto na percentagem do uso de processador e na quantidade de entradas do ficheiro redo. Idealmente, a carga deveria estar distribuída de igual forma pelos dois nodos, no entanto, em sistemas dinâmicos como é o caso da AIDA isso não é facilmente atingível. Apesar de graficamente apenas duas métricas possuírem um valor mais elevado no nodo 2 do que no nodo 1, existem outras que são praticamente iguais nos dois nodos como é o caso da memória e dos rácios buffer cache e chamadas recursivas. Este desnivelamento acontece devido às aplicações por defeito acederem ao nodo 1, não obstante, já existem implementadas também muitas aplicações que acedem ao nodo 2 sendo que a tendência é a equilibrar a utilização nos dois nodos.

Derivado ao comportamento dinâmico da base de dados AIDA foram então calculados os limites superiores e inferiores ao longo do dia para cada métrica de forma a ser possível observar a tendência ao logo do dia. Na maioria das estatísticas o limite mais importante é o limite superior, pois na maior parte delas um valor mais elevado é um bom indicador de problemas.

Número de sessões

Na Figura 6.1, encontra-se o gráfico que representa a variação do **número de sessões** ao longo do dia no nodo chp-ora01. Através do gráfico da Figura 6.1 pode-se observar os limites superior (p75) e inferior (p25) do número de sessões em cada intervalo de tempo. É importante salientar, que 50% dos dados recolhidos encontram-se neste intervalo, sendo expectável que as próximas medições também se encaixem dentro deste intervalo. Através da visualização do gráfico da Figura 6.1 é possível verificar que, normalmente, no período compreendido entre as 09h00 e as 20h00 existe um maior número de sessões, observando-se um pico máximo de sessões no intervalo das 12h00 às 13h00. De acordo com a teoria dos percentis, neste intervalo, existe 50% de probabilidade do número de sessões se encontrar entre as trezentas e setenta e cinco e as quinzentas sessões.

Na Figura 6.2, encontra-se o gráfico relativo aos limites do **número de sessões** do nodo 2 da AIDA. É interessante verificar que a distribuição neste nodo segue a mesma tendência do nodo 1 e até são coincidentes no intervalo de pico. No entanto, é necessário salientar que os valores dos percentis do nodo 2 são ligeiramente inferiores ao do nodo 1, o que pode ser justificado pelo um menor número de aplicações associadas ao nodo 2.

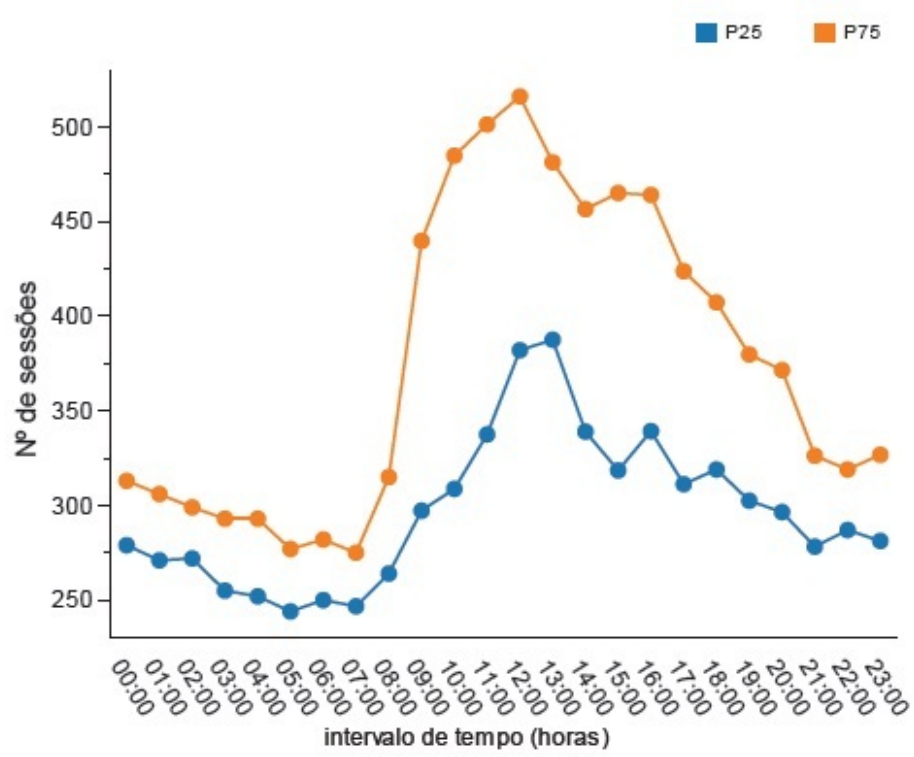


Figura 6.1: Gráfico do número de sessões ao longo do dia (chp-ora01)

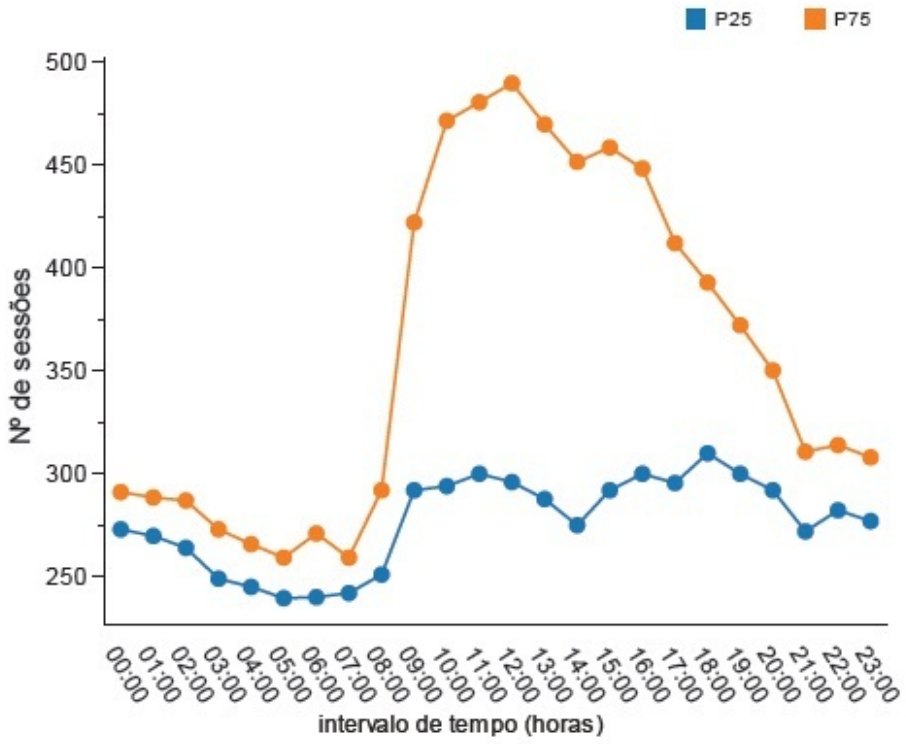


Figura 6.2: Gráfico do número de sessões ao longo do dia (chp-ora02)

Percentagem de utilização do processador

Através da consulta do gráfico da Figura 6.3, é possível observar uma maior percentagem de utilização de processador na parte da manhã, mais concretamente entre as 10h00 e as 13h00 com os valores a rondar os 30% de utilização. É importante salientar que aqui está apenas representada a percentagem de utilização do processador relativa aos processos dos utilizadores, este valor será incrementado se a percentagem de uso por parte dos processos do sistema for considerada, o que resultará num resultado final substâncialmente superior.

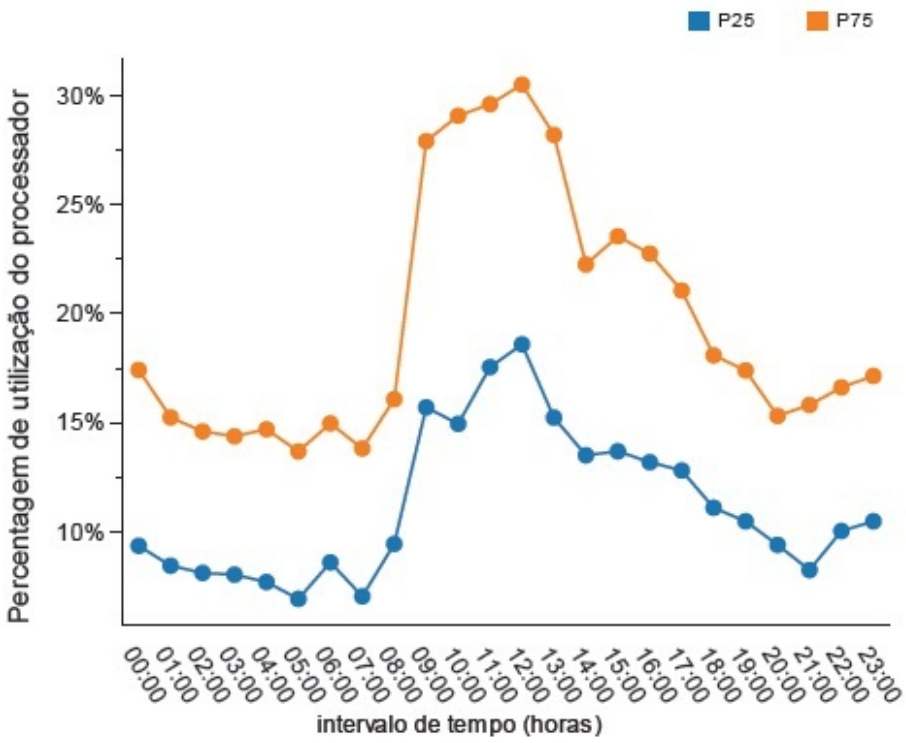


Figura 6.3: Gráfico da percentagem de utilização do processador ao longo do dia (chp-ora01)

A variação dos limites para esta métrica é praticamente igual nos dois nodos. Como se pode observar na Figura 6.4, o comportamento é praticamente igual diferindo apenas na parte final da noite onde os valores de processador decrescem no nodo 2, atingindo até valores abaixo dos 10% enquanto que no nodo 1 apresenta uma ligerira subida até aos 17,5%.

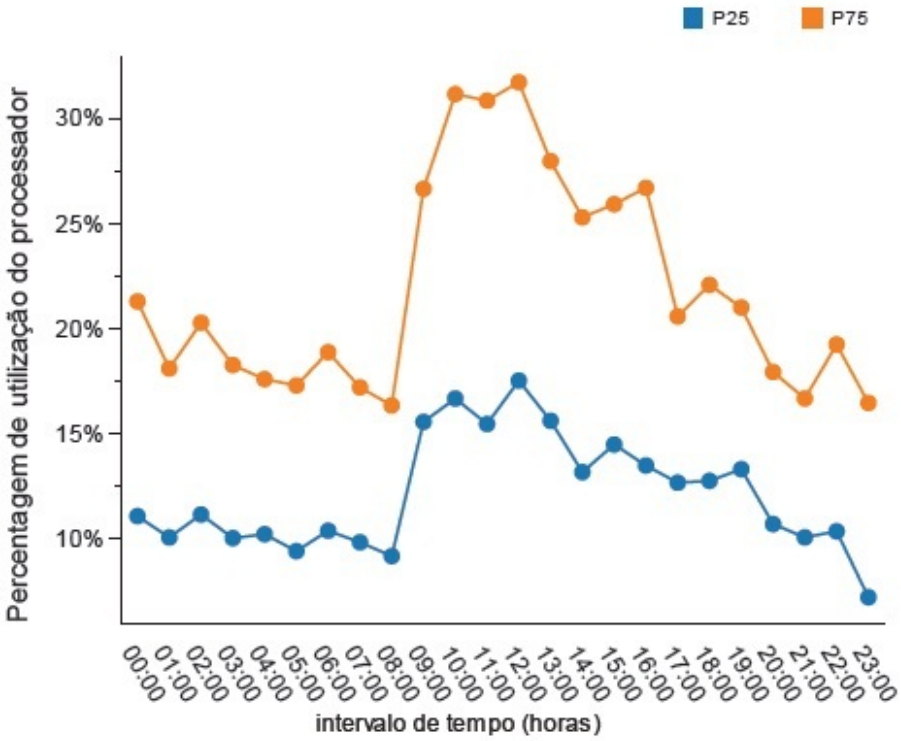


Figura 6.4: Gráfico da percentagem de utilização do processador ao longo do dia (chp-ora02)

Percentagem de utilização de memória

A **utilização de memória**, no nodo1, assume valores elevados ao longo de todo o dia, como se pode visualizar na Figura 6.5, sendo que, a altura mais crítica situa-se entre 9h00 e as 12h00. Neste intervalo, existe 50% de probabilidade dos valores se encontrarem entre 98,25% e 99,5%. Estes elevados valores de memória requerem bastante atenção, uma vez que, a memória é um fator que influência muito a rapidez e consequente disponibilidade da base de dados.

Distanciando-se das métricas até aqui apresentadas, a utilização de memória não é similar nos dois nodos. Apesar de no nodo 2 os valores de utilização de memória serem também elevados ao londo do dia a distribuição desses valores é diferente da do nodo 1. Através da Figura 6.6, é possível observar que no nodo 2 da 01h00 às 06h00 da manhã a percentagem de utilização de memória ultrapassa os 99%. Este valor tão elevado, associado ao percentil 75, só é obtido novamente durante o dia no intervalo das 14h00. Esta variação indica a presença de alguns processos pesados que são executados durante a noite para não prejudicar o bom funcionamento da base de dados numa altura de maior

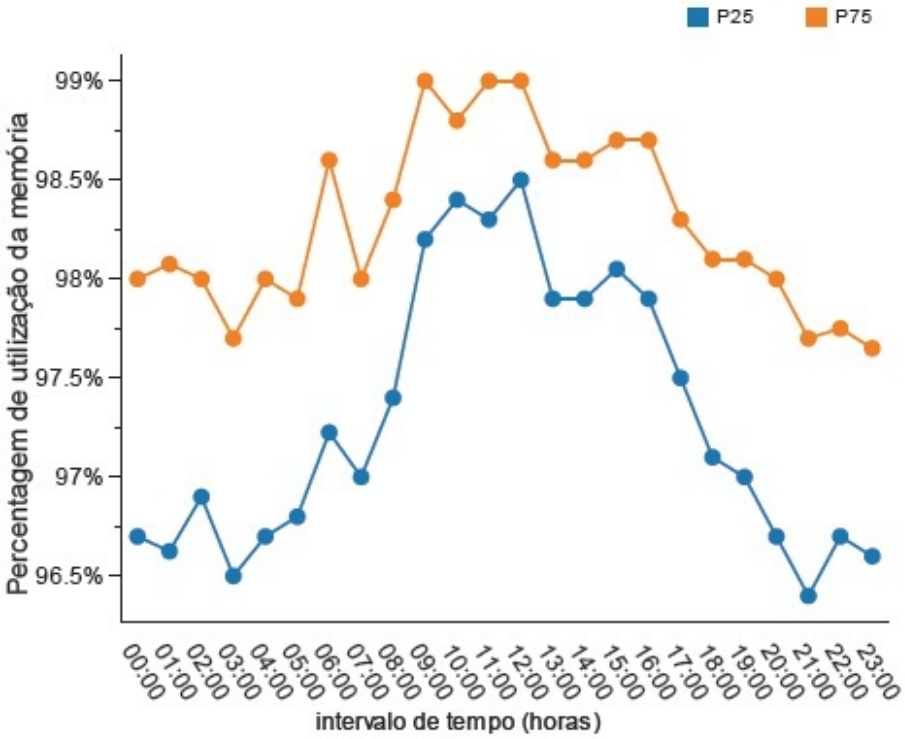


Figura 6.5: Gráfico da percentagem de utilização de memória ao longo do dia (chp-ora01)

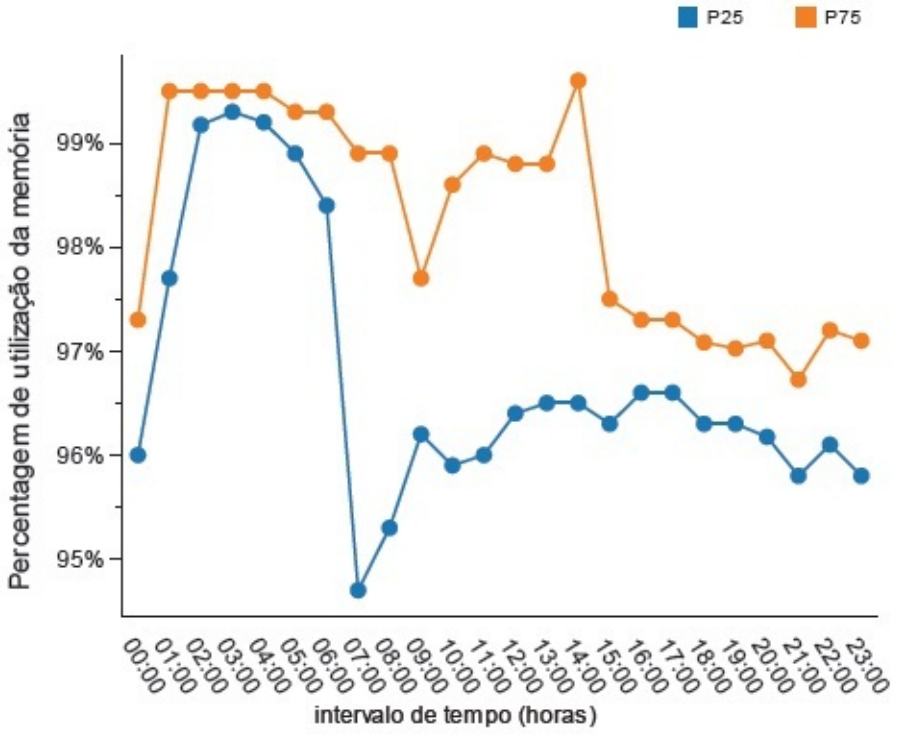


Figura 6.6: Gráfico da percentagem de utilização de memória ao longo do dia (chp-ora02)

utilização.

Volume tráfego de rede

Como se pode verificar na Figura 6.7, esta métrica apresenta no nodo 1 uma distribuição em "forma de sino", semelhante à das métricas relativas às sessões, ao processador e à memória. Apresenta menores valores durante o fim da tarde, à noite e durante a madrugada. O período onde é o volume de tráfego de rede é mais elevado situa-se no intervalo das 08h00 às 20h00. Este período coincide quase por inteiro com o do número de sessões, o que já era de prever, uma vez que a maior parte das sessões fazem uso da rede para enviar os seus pedidos, desta forma é expectável que quando maior o número de sessões maior a utilização da rede. Normalmente os valores máximos registam-se no intervalo das 9h00 as 13h00 estando o percentil 75 na ordem dos 600000 bytes por segundo.

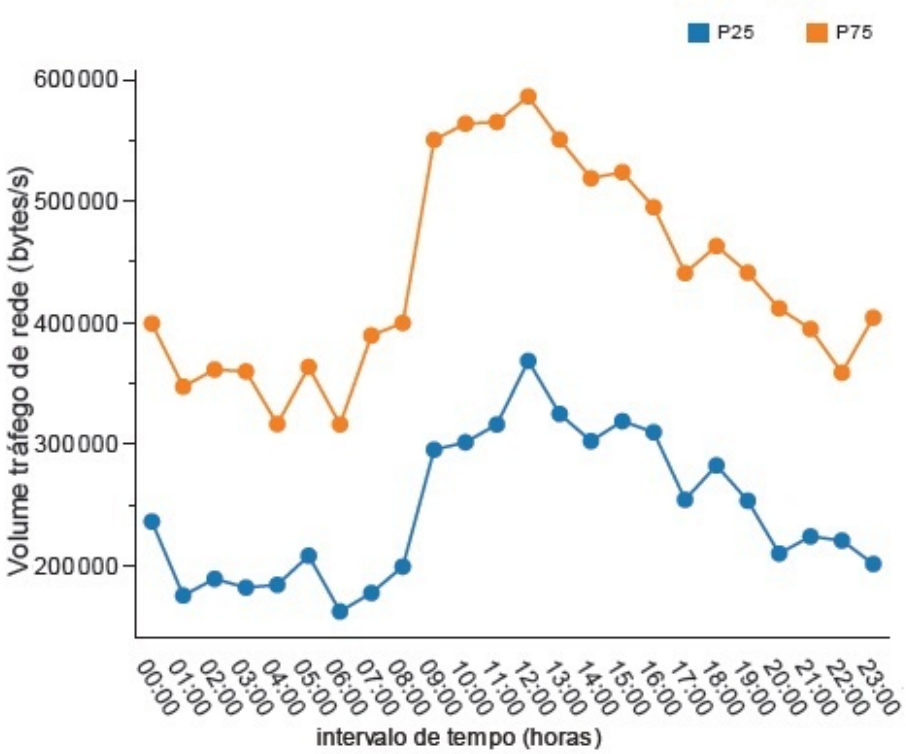


Figura 6.7: Gráfico do volume do tráfego de rede ao longo do dia (chp-ora01)

Relativamente ao comportamento desta métrica mas no nodo 2, este é similar ao do nodo 1. No entanto, verifica-se através da Figura 6.8, uma ligeira diminuição dos valores dos percentis. Apenas no intervalo das 12h00, o percentil 75 é superior aos 600000 bytes por segundo.

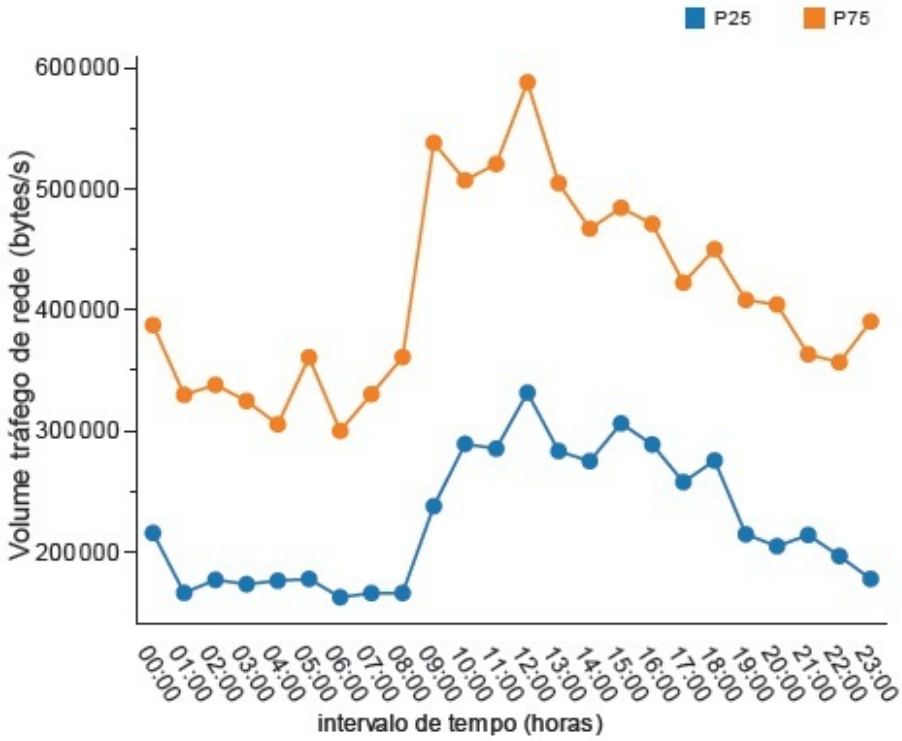


Figura 6.8: Gráfico do volume do tráfego de rede ao longo do dia (chp-ora02)

DB time

Como já foi visto anteriormente o DB time pode ser visto como uma medida para a avaliação do tempo de resposta da base de dados. Sendo assim, quanto maior o DB time maior é o tempo que demora a execução de operações na base de dados. Na Figura 6.9, são apresentados os limites de normalidade para o nodo 1 da AIDA. É possível observar valores mais elevados no período compreendido entre as 9h00 e as 14h00, fruto da maior carga de trabalho que a base de dados está sujeita durante este período. No entanto, por volta das 22h00 encontra-se outro pico no valor desta métrica relacionado com processos de *background* que são executados por volta desta hora.

No nodo 2, como se pode observar na Figura 6.10, as curvas dos dois percentis apresentam valores um pouco inferiores dos verificados no nodo 1, mas os períodos críticos coincidem, ou seja, globalmente os dois nodos apresentam a mesma capacidade de resposta durante os mesmos períodos de tempo.

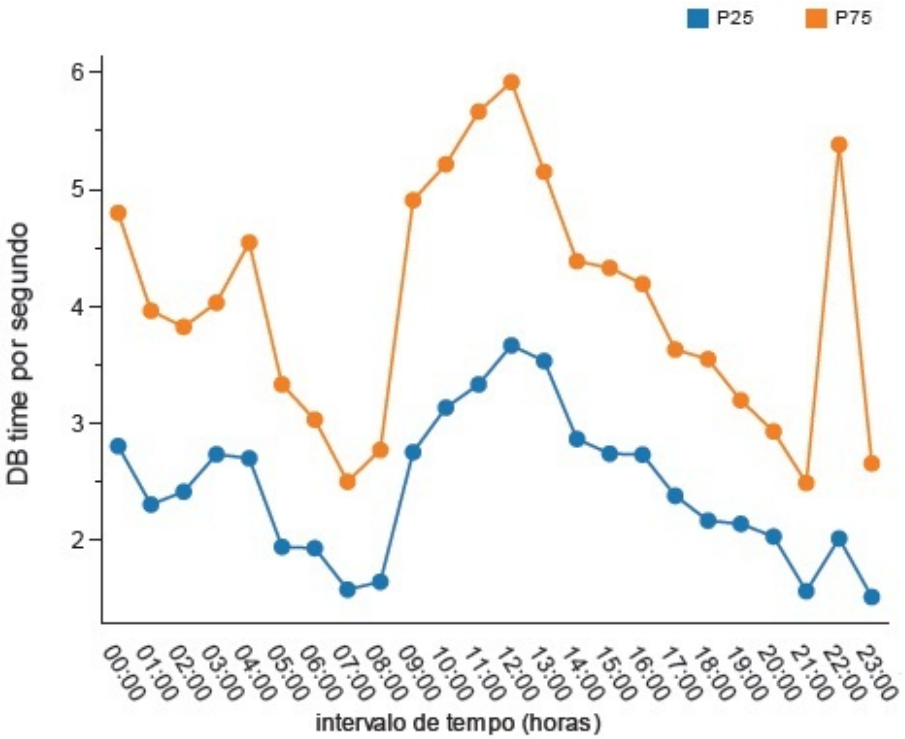


Figura 6.9: Gráfico do db time ao longo do dia (chp-ora01)

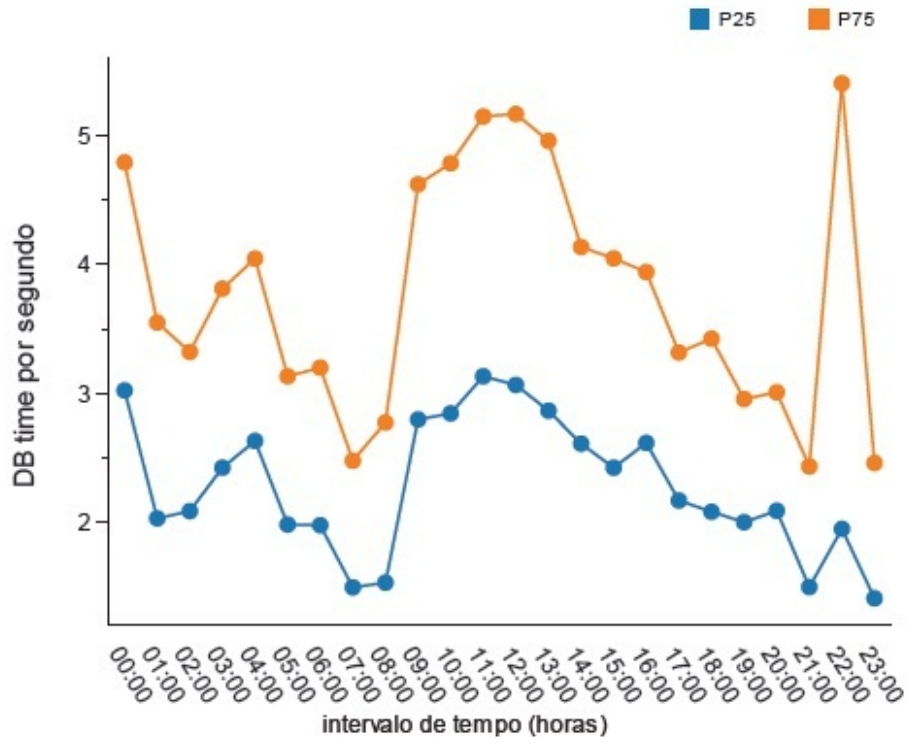


Figura 6.10: Gráfico do db time ao longo do dia (chp-ora02)

Rácio Buffer Cache

O **Rácio do Buffer da Cache** é uma variável um pouco diferente das restantes, devido a neste caso, o limite mais importante é o inferior pois baixos valores deste rácio é que podem indicar problemas de memória. Devido à robusteza física das bases de dados do CHP este rácio tem pouca variabilidade, apresentando quase sempre valores perto de 1 (valor ideal). No entanto, é possível identificar alguns intervalos onde a performance não é tão satisfatória. Como se pode observar na Figura 6.11, existe um decréscimo no valor do rácio por volta das 01:00, sendo que o percentil 25 caí para baixo de 0,985. Derivado à altura da noite em que ocorre este pico é originado por rotinas de backup que acontece por volta desta hora. Outras diminuições do rácio são visíveis por volta das 05h00 e 22h00, mas estas não são tão acentuadas.

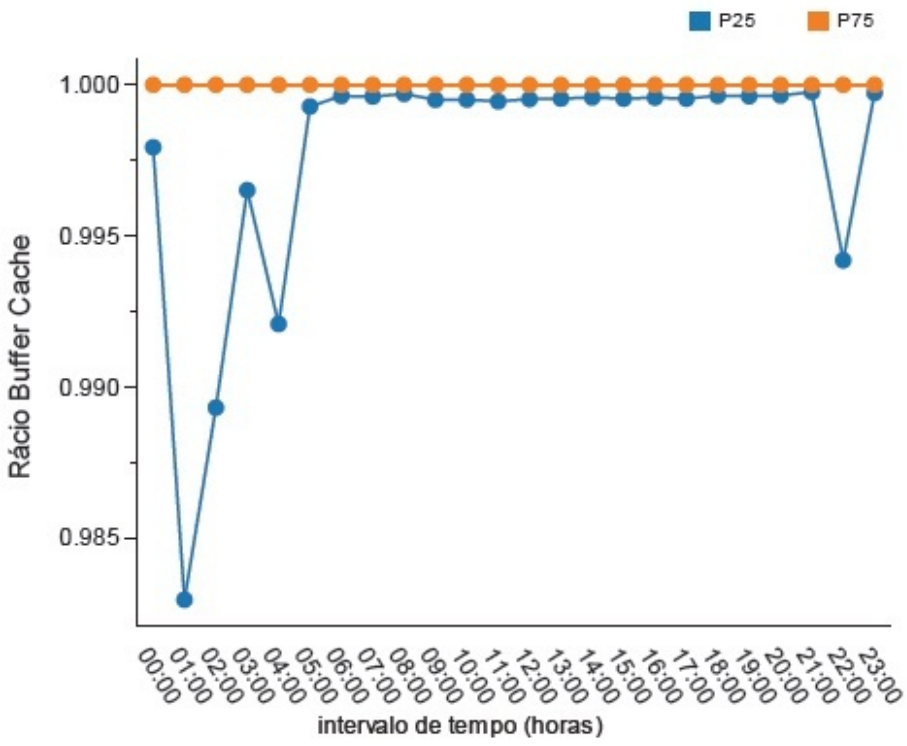


Figura 6.11: Gráfico do rácio buffer cache ao longo do dia (chp-ora01)

No nodo 2, os horários dos picos coincidem, no entanto, como se pode observar na Figura 6.12, as descidas não são tão elevadas. Como todos os valores se situam na ordem dos 0,99 estes picos não têm grande impacto para a deteção de problemas relacionados com a memória, pois as descidas não são significativas.

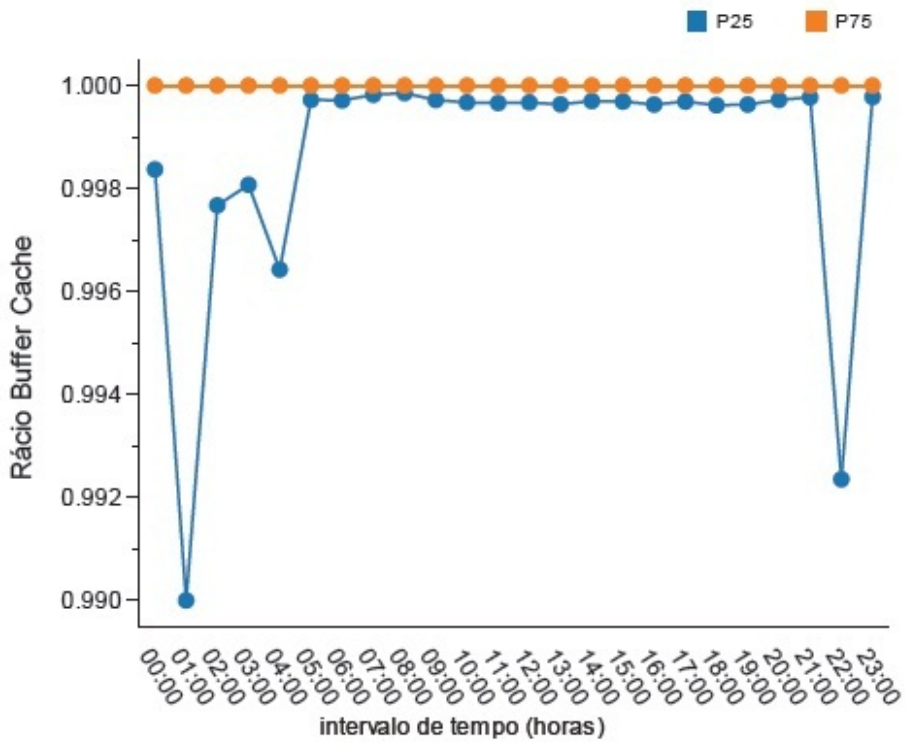


Figura 6.12: Gráfico do rácio buffer cache ao longo do dia (chp-ora02)

Rácio de chamadas recursivas

Na Figura 6.13, encontra-se o gráfico representativo da variação dos limites para o rácio de chamadas recursivas no nodo 1. Curiosamente, as alturas em que são apresentados valores mais elevados para este rácio são as mesmas onde o rácio buffer cache é menos elevado, ou seja, estão relacionados com os processos de sistema que são executados a estas horas. Ao longo do dia, o percentil 75 é relativamente baixo, no entanto, no intervalo da 01h00 o valor do percentil é superior a 0,6, um valor já bastante elevado.

Relativamente ao nodo 2, como se pode observar na Figura 6.14, os limites do rácio apresentam um padrão similar ao do nodo 1 e mais uma vez apenas diferem no valor dos limites verificando-se que os valores no nodo 2 são mais baixos. Neste nodo, existe também um pico no intervalo das 00h00 aumentando o intervalo de tempo a ter em conta para a avaliação deste rácio.

Número de pedidos de Input/Output

Os limites do número de pedidos Input/Output, como se pode observar na Figura 6.15 , têm uma tendência muito similar às anteriores. Verifica-se um pico no intervalo das

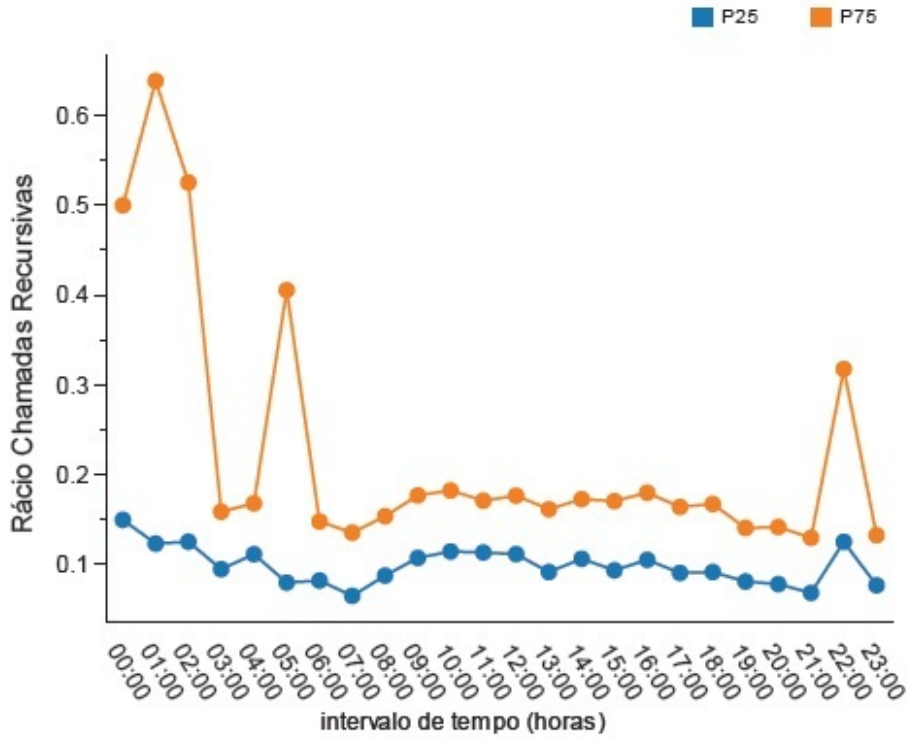


Figura 6.13: Gráfico do rácio das chamadas recursivas ao longo do dia (chp-ora01)

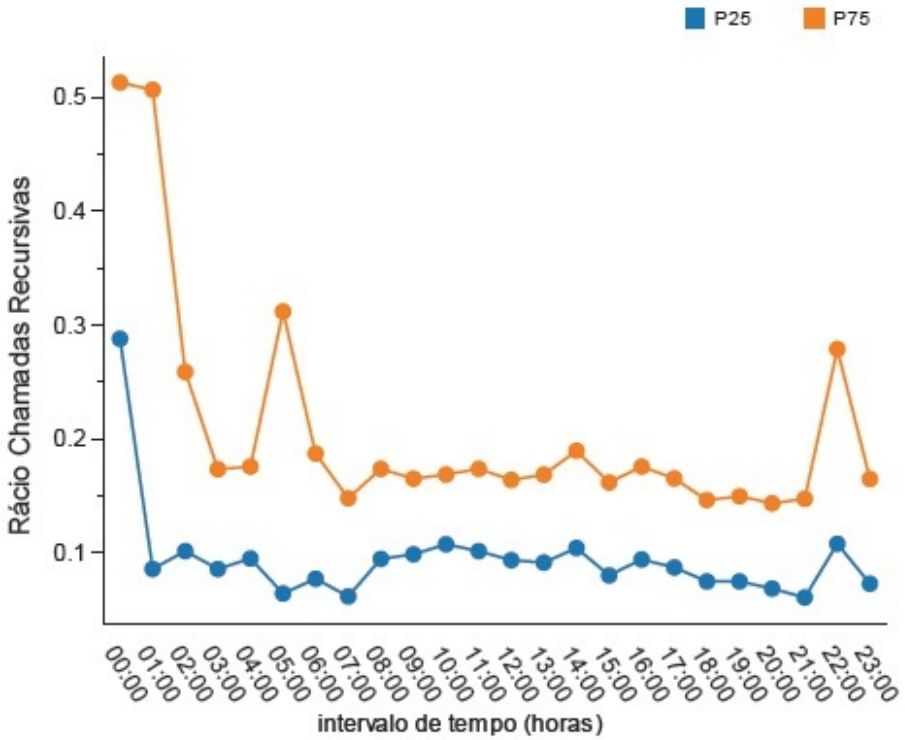


Figura 6.14: Gráfico do rácio das chamadas recursivas ao longo do dia (chp-ora02)

01h00, com 50% dos valores compreendidos entre 200 e 1200 pedidos por segundo. Às 22h00 horas apresenta-se também um pico mas de menor amplitude, sendo que o percentil 75 está situado por volta dos 600 pedidos por segundo. Nos restantes períodos do dia os limites mantêm-se constantes com percentil 25 a rondar o valor de 200 e o percentil 75 o valor de 400 pedidos por segundo. Esta métrica é muito influênciada pela ocorrência de backups.

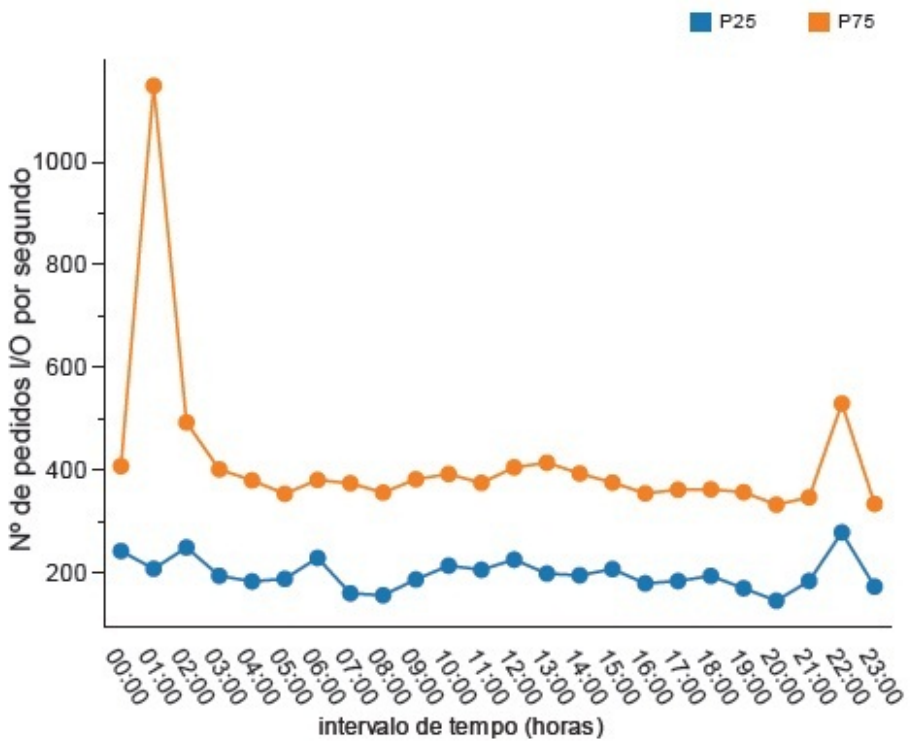


Figura 6.15: Gráfico do número de pedidos de I/O longo do dia (chp-ora01)

Através da Figura 6.16, pode-se verificar que no nodo 2 o comportamento é similar. No entanto, o valor do pico da 01h00 é cerca de metade do verificado no nodo 1. Relativamente ao pico das 22h00, este é ligeiramente mais pequeno do que o pico do nodo 1. Nos restantes períodos do dia a variabilidade no nodo 2 é mais acentuada, no entanto, excetuando o intervalo das 06h00 não mais o percentil 75 ultrapassa os 400 pedidos por segundo.

Número de transações

Os limites relativos ao número de **transações por segundo** apresentam uma grande variabilidade de intervalo para intervalo mas numa escala reduzida. Como se pode observar

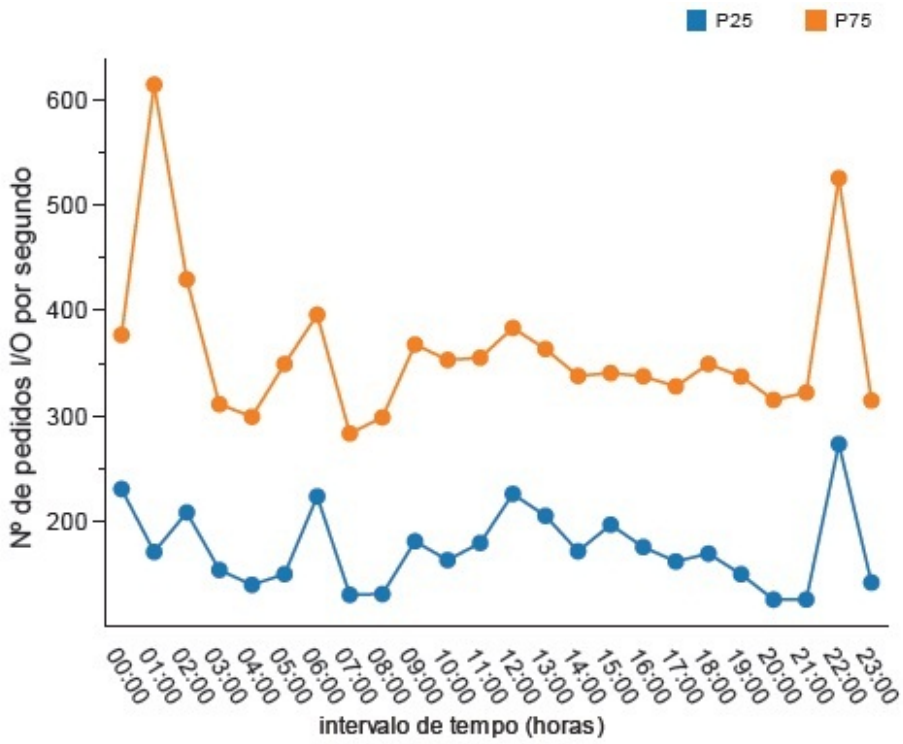


Figura 6.16: Gráfico do número de pedidos de I/O longo do dia (chp-ora02)

na Figura 6.17, o percentil 75 está situado quase sempre entre as 140 e 160 transações por segundo, exceção feita por volta das 04h00, 06h00 e 22h00 onde o valor do percentil 75 é um pouco mais baixo. É importante salientar que o intervalo definido pelos dois percentis é muito grande, o que indica uma substancial diferença de valores obtidos no conjunto das medições. Desta forma, pode-se constatar que esta estatística varia mais de dia para dia do que de hora para hora ao longo do mesmo dia.

O mesmo se pode dizer acerca do nodo 2, onde a diferença entre o limite definido pelo percentil 25 e o limite definido pelo percentil 75 é também elevada. Neste caso, como se pode visualizar na Figura 6.18, o valor do percentil 75 situa-se entre as 120 e 150 transações por segundo, não apresentando exceções. Devido às transações serem consideradas as unidades de trabalho das bases de dados, pode-se verificar aqui que o nodo 2 da base de dados AIDA tem uma menor carga de trabalho em virtude do número de transações.

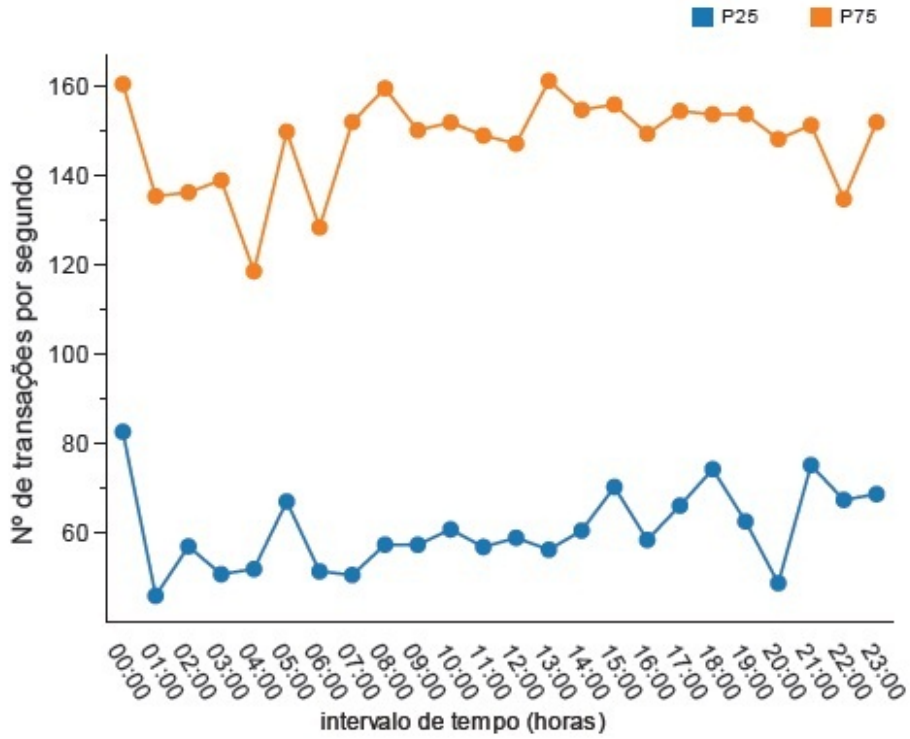


Figura 6.17: Gráfico do número de transações por segundo ao longo do dia (chp-ora01)

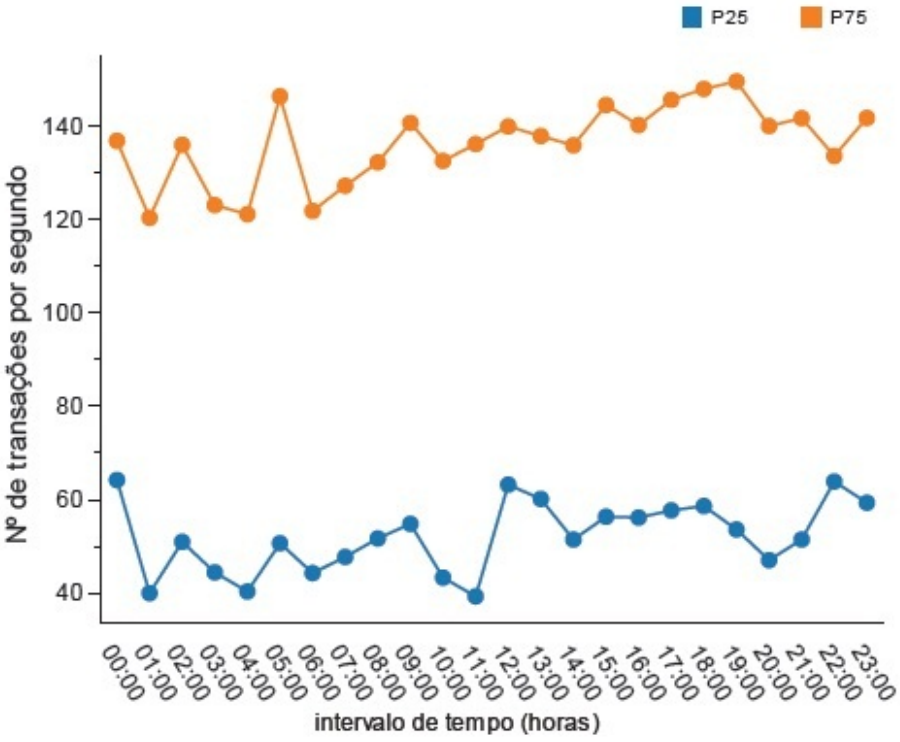


Figura 6.18: Gráfico do número de transações por segundo ao longo do dia (chp-ora02)

Número de Operações

Os limites do **número de operações por segundo** possuem uma distribuição particular que pode ser observada na Figura 6.19. Apresentam picos elevados por volta do intervalo das 00h00 e das 02h00 e depois durante o dia, das 08h00 às 20h00 apresentam uma curva em "forma de sino" mas de muita menor amplitude da registada nos picos. Por exemplo, mo intervalo das 02h00 o valor associado ao percentil 75 é de cerca de 1200 operações por segundo, enquanto que no ponto mais alto da curva em "forma de sino", o valor é apenas de 600 operações por segundo. Outro aspecto a salientar, é o elevado número de operações que uma transação pode exigir. Como foi visto anteriormente, o número de transações ronda, no máximo, as 160 transações por segundo, sendo que o número de operações, mesmo excluindo as situações de pico, é algumas centenas superior, o que torna esta estatística também muito importante para avaliar a carga de trabalho da base de dados.

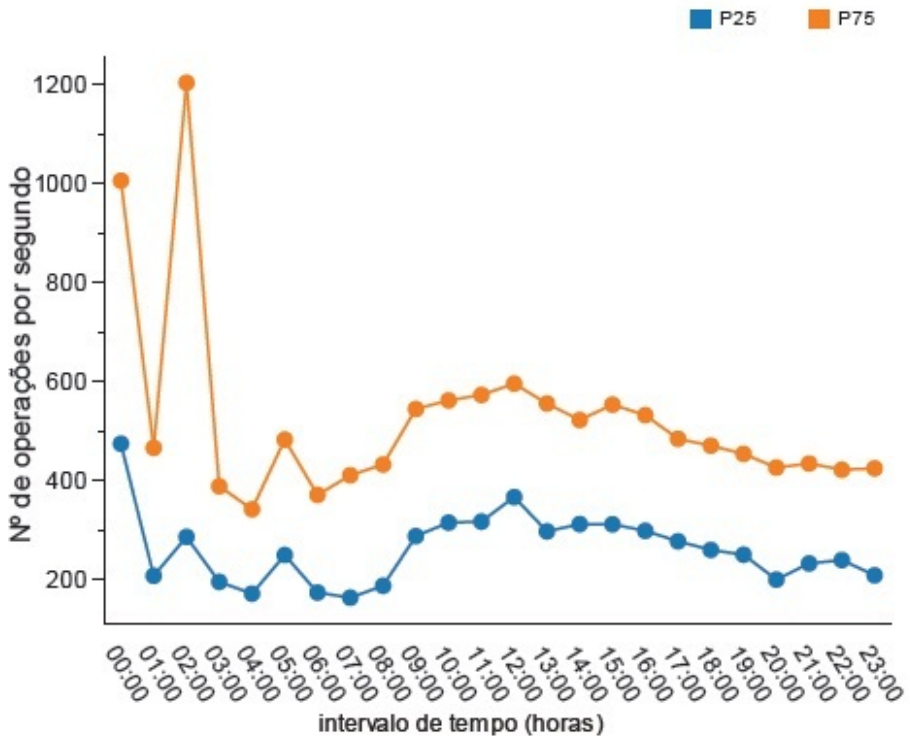


Figura 6.19: Gráfico do número de operações por segundo ao longo do dia (chp-ora01)

Em relação aos limites do nodo 2, é possível observar na Figura 6.20, que existe apenas um pico às 00h00 onde o intervalo definido pelos percentis vai de cerca de 450 a 1050. Ao contrário do nodo 1, neste nodo não existe pico no intervalo das 02h00. Nos

restantes períodos, regista-se um aumento do número de operações por volta das 09h00 mantendo-se até às 16h00 com o percentil 75 acima dos 500. A partir das 16h00 nota-se uma ligeira diminuição até se estabilizar nas 400 operações por segundo até ao fim do dia. De igual forma, ao que acontece no número de transações, a ordem de grandeza das operações por segundo também diminuí no nodo 2.

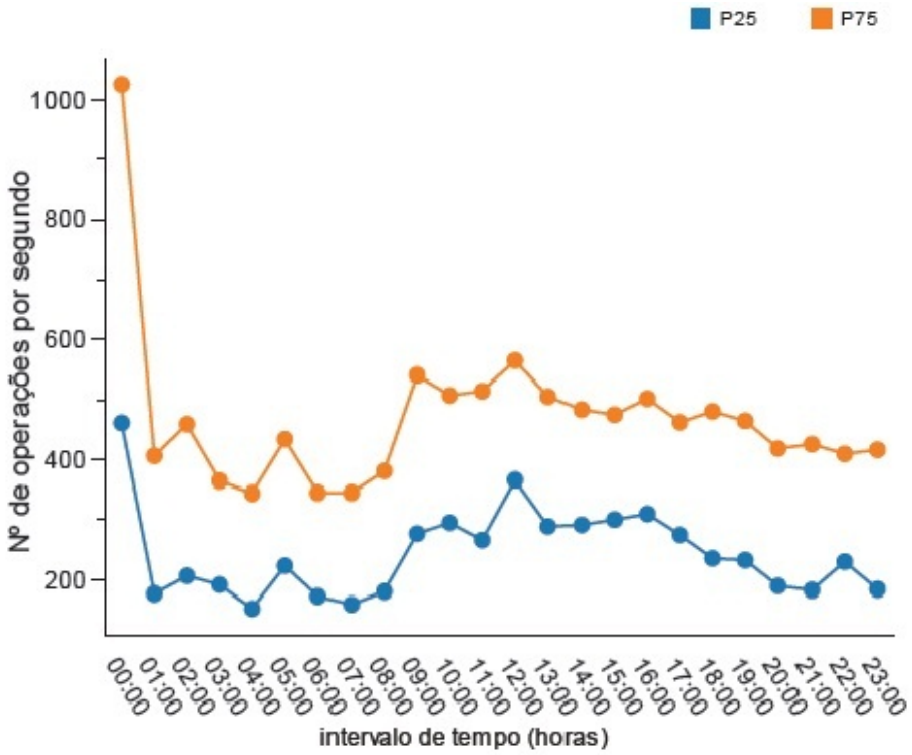


Figura 6.20: Gráfico do número de operações por segundo ao longo do dia (chp-ora02)

Redo size

Na Figura 6.21, encontra-se o gráfico representativo dos limites desta variável expressa em kb/s. Esta variável apresenta um pico de maior amplitude no intervalo das 00h00 onde 50% encontram-se compreendidos entre os 0 kb/s e os quase 300 kb/s. Nos restantes intervalos de tempo, os limites são práticamente constantes sendo o limite superior cerca de 100 kb/s. A par do que acontece com o número de operações de I/O, esta métrica também é muito influenciada por rotinas de backup.

Relativamente aos limites do nodo 2 estes seguem a mesma distribuição do nodo 1. No entanto, o valor associado ao intervalo das 00h00 é maior. Neste intervalo, 50% dos dados recolhidos estão compreendidos entre 0 e 800 kb/s. Nos restantes períodos do dia

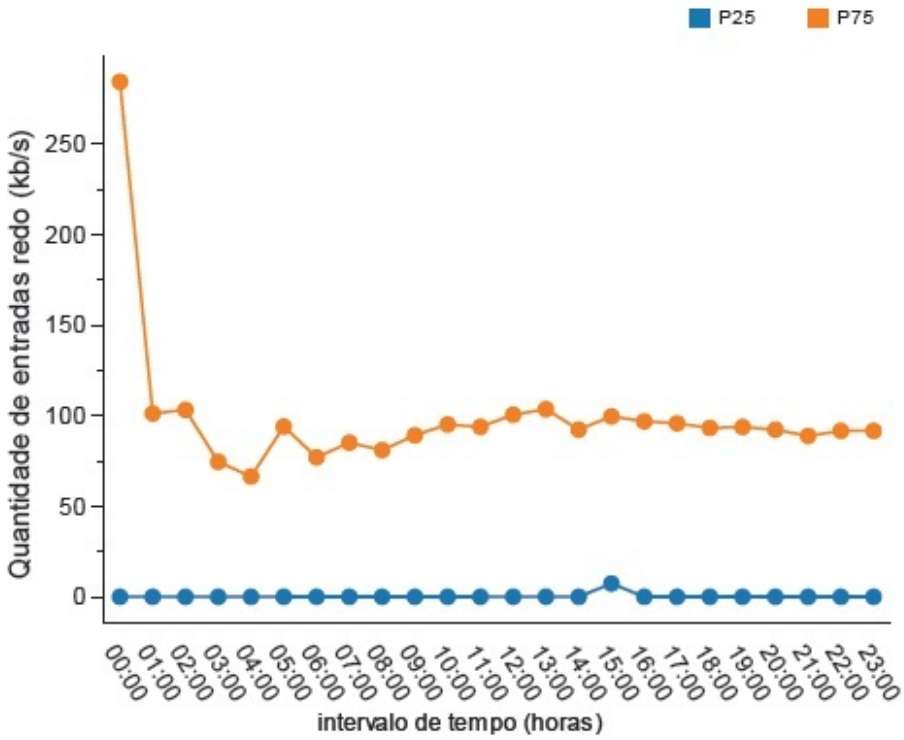


Figura 6.21: Gráfico da quantidade de redo size ao longo do dia (chp-ora01)

os limites assumem valores próximos dos limites do nodo 1, por isso, verifica-se que o período das 00h00 é o que influencia mais a média desta métrica contibuindo para que a média do redo size seja superior no nodo 2, como já foi verificado na Tabela 6.1.

Pedidos de espaço buffer redo log

Esta métrica apresenta valores extremamente baixos pois em bases de dados bem dimensionadas raramente existe falta de espaço para escrever no *buffer redo* pois é bem efetuada a gestão do mesmo. No entanto, é possível observar a partir da Figura 6.23, que os limites seguem uma tendência muito similar ao verificado no *redo size* mas com uma ordem de grandeza bastante inferior, como seria de esperar até porque as unidades em que se encontram representados são diferentes. No intervalo das 00h00, os valores encontram-se entre os 0 e os 0,08 pedidos por segundo e nos restantes períodos do dia o percentil 75 não é superior a 0,01.

Este é também o comportamento verificado no nodo 2, não existindo nenhumas diferenças significativas. Devido aos diminutos valores que esta métrica apresenta poderá se pensar que esta não é um bom indicador, no entanto, é de grande valor uma vez que um

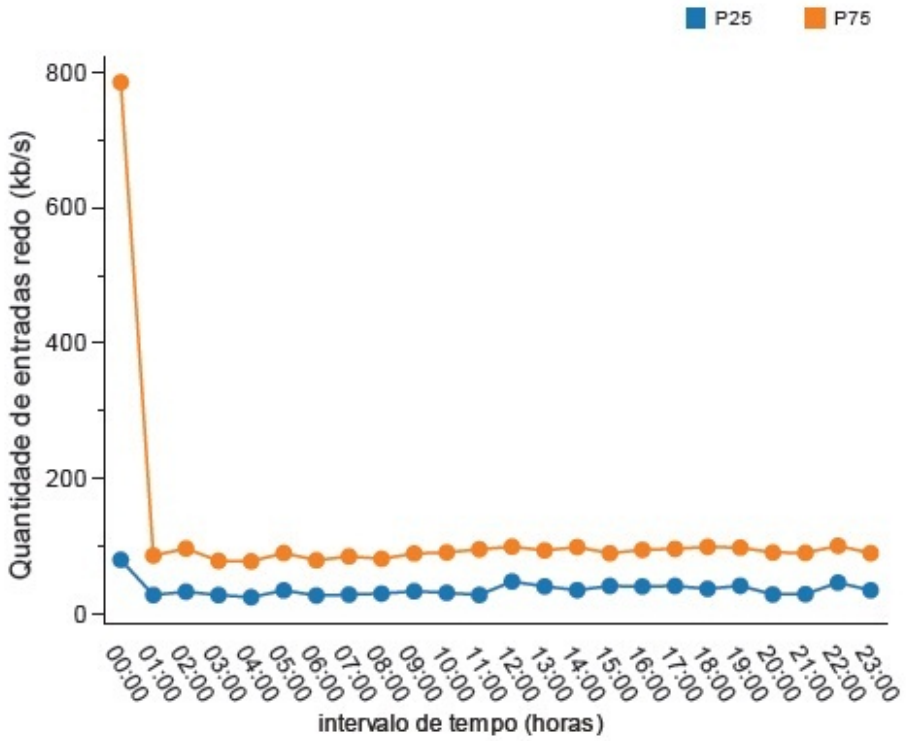


Figura 6.22: Gráfico da quantidade de redo size ao longo do dia (chp-ora02)

elevado número de pedidos pode provocar atrasos na transações e comprometer a função de *recovery* da base de dados perante uma possível falha.

Avaliação de picos

Para determinar os períodos temporais mais susceptíveis a falhas foi elaborado o *dashboard* aida_picos.wcdf. Na Figura 6.24, encontra-se um excerto desse *dashboard* relativo ao nodo chp-ora01. Como se pode observar o intervalo das 11h00 é aquele que possui mais picos em simultâneo. É importante salientar ainda que as métricas que atingem os picos neste intervalo são umas das mais importantes(uso de processador, memória, rede e número de sessões). Para além disso, às no intervalo das 12h00 são atingidos mais três picos de outras métricas. Outro período a salientar é o das 00h00 e da 01h00 estes mais relacionados com as estatísticas de *redo size*, e pedidos de espaço para o *buffer redo*. Este *dashboard* é atualizado diariamente o que significa que pequenas mudanças podem ocorrer de dia para dia. No entanto, verificou-se várias vezes que os períodos mais críticos situam-se nos intervalos: 11h00, 12h00, 00h00 e 01h00. Nestas alturas, a atenção deve ser redobrada no nodo chp-ora01.

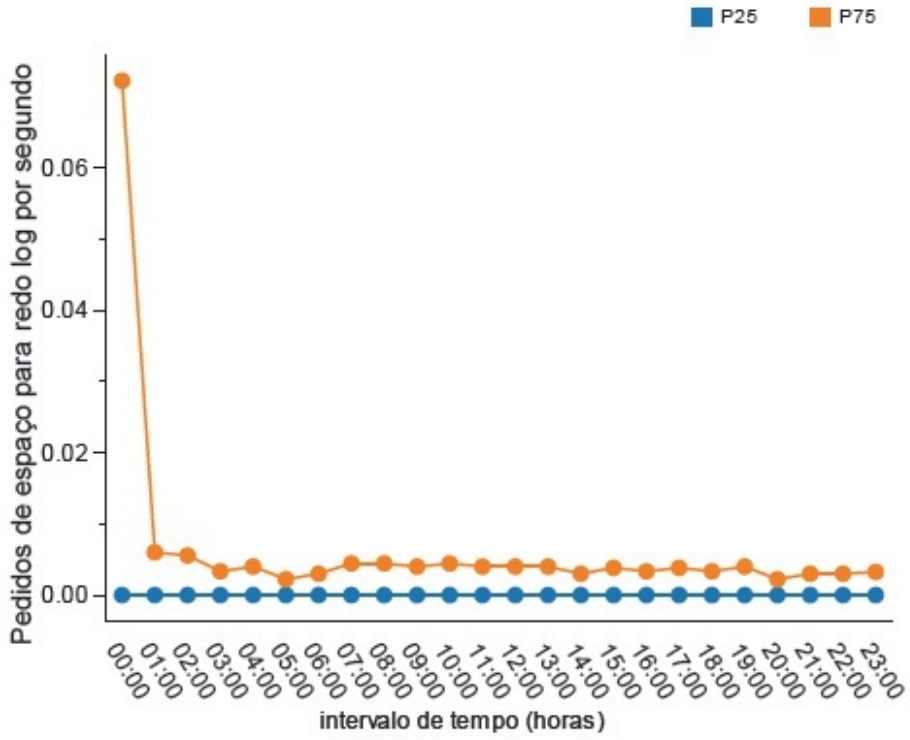
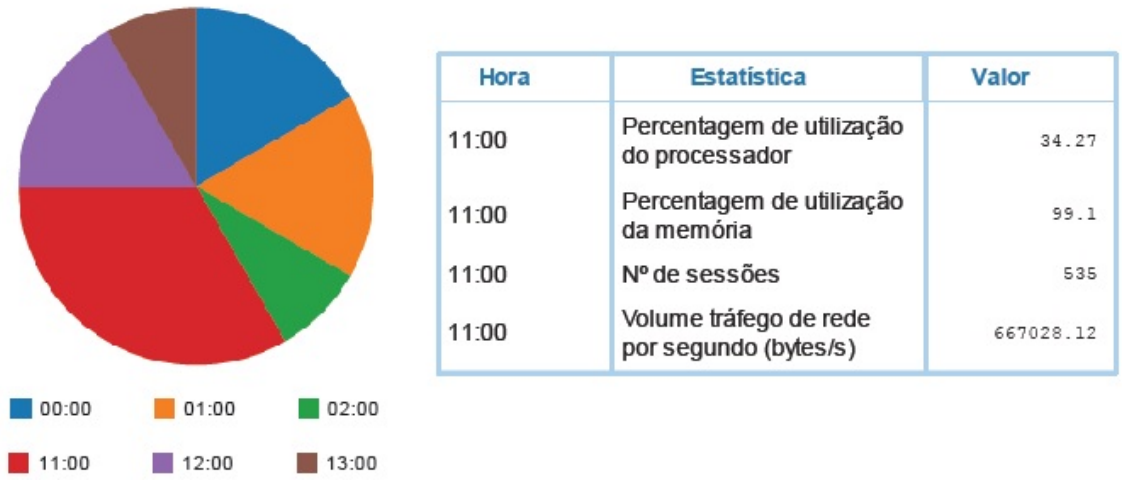


Figura 6.23: Gráfico do número pedidos de espaço para redo buffer por segundo ao longo do dia (chp-ora01)

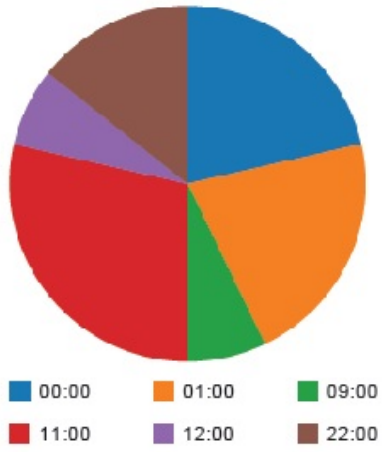


Hora	Estatística	Valor
11:00	Percentagem de utilização do processador	34.27
11:00	Percentagem de utilização da memória	99.1
11:00	Nº de sessões	535
11:00	Volume tráfego de rede por segundo (bytes/s)	667028.12

Figura 6.24: Excerto dashboard aida_picos.wcdf relativo ao chp-ora01

Relativamente ao nodo chp-ora02, os intervalos onde se registam maior número de picos em simultâneo são os mesmos relativos ao nodo chp-ora01. No entanto, existem mudanças ao nível das métricas associadas a cada intervalo temporal. Como se pode observar na Figura 6.25, no intervalo das 11h00 verificam-se picos relativos ao DB time, à percentagem de uso do processador e ao número de sessões. O número de sessões aparece duas vezes, significando que durante o período das 11h00 às 12h00 registaram-se duas vezes picos com igual valor.

Globalmente é possível afirmar que os períodos temporais críticos para a base de dados AIDA são os períodos compreendidos entre as 11h00 e as 12h00, bem como o período entre a 00h00 e a 01h00.



Hora	Estatística	Valor
11:00	Percentagem de utilização do processador	33.49
11:00	DB time por segundo	5.61
11:00	Nº de sessões	501
11:00	Nº de sessões	501

Figura 6.25: Excerto dashboard aida_picos.wcdf relativo ao chp-ora02

6.1.2 Base de dados SONHO

Como já foi mencionado anteriormente, devido ao facto da base de dados SONHO possuir a versão Oracle 7 esta ainda não possui as *performance views* relacionadas com as métricas DB time e volume de tráfego de rede, sendo que a análise foi elabora apenas tendo em conta as restantes dez métricas.

Para as métricas assinaladas procedeu-se ao calculo da média e do desvio padrão. Como se pode observar na Tabela 6.2, o desvio padrão é elevado para quase todas as métricas excetuando o rácio buffer cache. As razões para este fenómeno são as mesmas da base de dados AIDA: a variabilidade dos valores ao longo do dia, e o aumento sistemático de utilização da base de dados ao longo dos dias. Os rácios apresentam menores valores de

desvio padrão devido a variarem entre 0 e 1, desta forma, não são registados valores muito distantes da média, como é o caso das operações. Relembrando que as operações provêm das transações, existem transações simples que requerem poucas operações e transações complexas que envolvem um grande número de operações, estas diferenças aumentam bastante o desvio em relação à média.

Ainda com base na Tabela 6.2, é possível verificar que a base de dados SONHO tem grande utilização pois a média do número de sessões ronda as quinhentas e vinte. Relativamente à carga de trabalho, apesar de apresentar um número de transações por segundo bastante mais pequeno do que a AIDA, destas resultam em média 748 operações por segundo, ultrapassando até a média de operações da base de dados AIDA, podendo tal facto indicar uma maior complexidade das transações na base de dados SONHO. Outro facto a salientar, é o número mais reduzido de pedidos de I/O por segundo na base de dados SONHO do que na base de dados AIDA. Esta situação pode dever-se ao facto de as estatísticas utilizadas para o cálculo desta métrica não serem as exatamente as mesmas em virtude da versão do SONHO ser mais antiga.

Tabela 6.2: Resumo métricas relativas à carga de trabalho do SONHO

Nome	Média	Desv. Padrão
Nº de transações por segundo	10.264	8.768
Percentagem de utilização do processador	26.749	6.894
Memória Livre	2694795.011	24329.202
Nº de sessões	520.541	273.67
Nº de pedidos I/O por segundo	34.544	42.398
Nº de operações por segundo	747.893	1242.023
Rácio Buffer Cache	0.961	0.006
Quantidade de entradas redo (kb/s)	37.531	53.369
Rácio Chamadas Recursivas	0.379	0.167
Pedidos de espaço para redo log por segundo	0.001	0.002

De seguida apresentam-se os gráficos representativos do comportamento padrão das métricas selecionadas da base de dados SONHO. A análise realizada é análoga à efetuada para a base de dados AIDA. Foram calculados os limites superiores (através do percentil 75) e inferiores (através do percentil 25) para cada intervalo de tempo, neste caso em cada hora.

Número de sessões

Com base na Figura 6.26, pode-se afirmar que o número de sessões na base de dados SONHO apresenta dois padrões diferentes. No intervalo das 08h00 às 21h00, o limite superior é bastante elevado, registando um valor máximo por volta das 12h00 onde 50% dos dados estão situados entre as 300 e cerca das 1100 sessões. No intervalo das 22h00 às 07h00 os limites encontram-se muito próximos, o que indica pouca variabilidade entre as medições. Para além disso, é possivel observar que ao longo deste intervalo 50% dos valores recolhidos rondam apenas as 300 sessões. A distribuição desta métrica mostra que esta base de dados é essencialmente utilizada ao durante o período diurno.

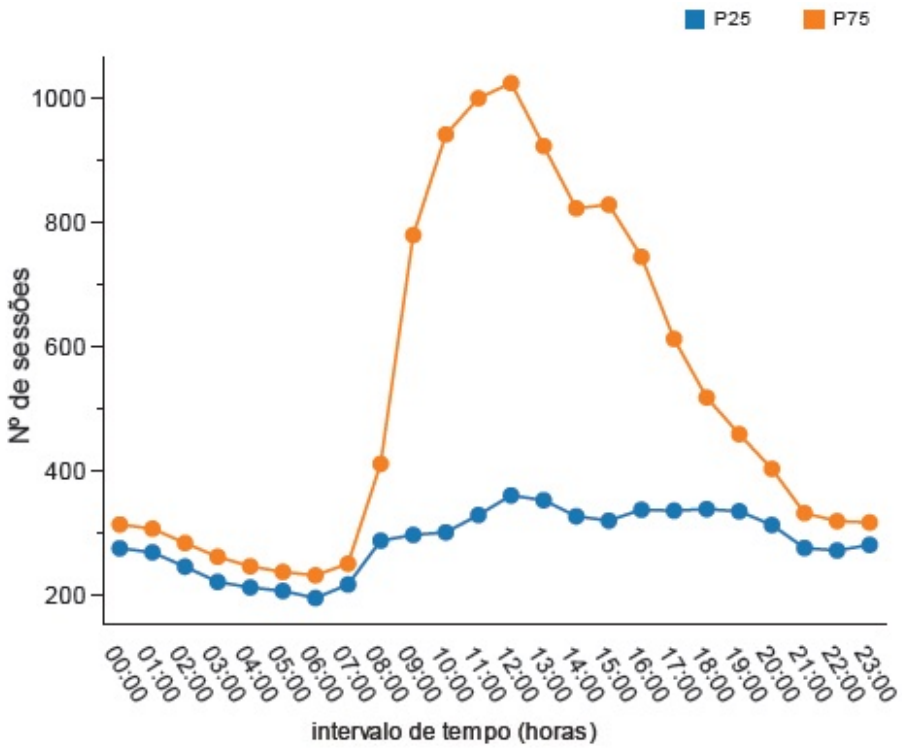


Figura 6.26: Gráfico do número de sessões ao longo do dia no SONHO

Percentagem de utilização do processador

Relativamente à utilização do processador, como se pode observar na Figura 6.27, a distância entre os dois limites é significativa o que indica uma variabilidade dos valores para esta métrica em cada intervalo de medição. Normalmente, esta variação está dependente do escalonamento de processos efetuado pelo sistema operativo. Existem três intervalos onde os limites superiores são substancialmente diferentes: das 01h00 às 04h00 e das

20h00 às 23h00 com o valor do percentil 75 a rondar praticamente os 30% de utilização; e o intervalo das 08h00 às 16h00 onde o valor so percentil 75 é substâncialmete maior, ultrapassa os 35% de utilização chegando em alguns casos a rondar os 40%. O intervalo no qual se regista um percentil 75 maior, é o intervalo das 12h00 onde 50% dos dados situam-se 19% e 40% de utilização do processador.

Mais uma vez importa salientar que esta percentagem é apenas relativa aos processos dos utilizador, permanentemente correm nos servidores também processos do sistema que gastam processador, no entanto, como o seu comportamento é homogéneo ao longo dos dias não foram aqui considerados. Note-se que os 40% de utilização por parte de processos de utilizador pode já ser uma situação perigosa, caso existam outros 40% de ocupação por processos do sistema.

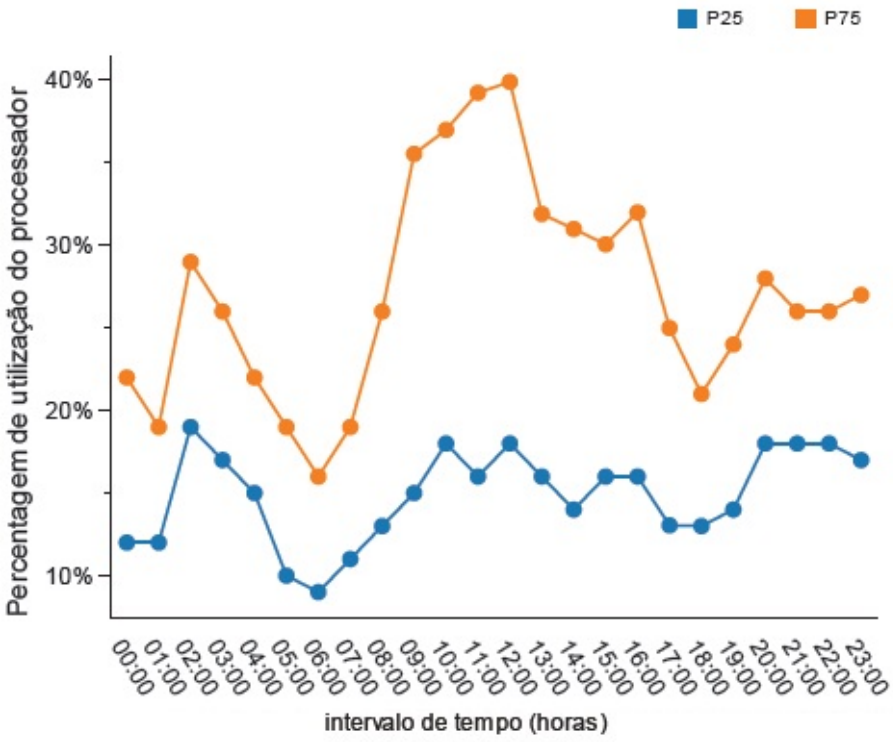


Figura 6.27: Percentagem de utilização de processador ao longo do dia no SONHO

Utilização de Memória

Como já foi visto anteriormente, as versões Oracle utilizadas nas bases de dados AIDA e SONHO são diferentes, mas para além disso também os sistemas operativos dos servidores são distintos. No servidor da base de dados AIDA é utilizado um sistema Linux,

enquanto que na base de dados SONHO é suportada por um servidor HP-UX. Esta diferença influenciou a escolha dos comandos a usar para recolher dados relativos à memória e uso do processador. Para a recolha dos dados relativos ao servidor da base de dados SONHO teve de ser utilizado o comando *vmstat*. Com este comando apenas foi possível recolher o número de memória livre disponível expressa em páginas de memória livres (uma página corresponde a 4 kilobytes). Estes valores podem ser algo incrementados pois também têm em conta a memória virtual.

Desta forma, na Figura 6.28 pode-se observar os limites da quantidade de páginas livres de memória ao longo do dia. É possível identificar o período entre as 8h00 e as 16h00 como aquele em que existe uma maior utilização de memória, indicada pela diminuição do limite representativo do número de páginas livres. No intervalo mais crítico deste período, o das 12h00 o limite inferior é cerca de quinhentas mil páginas mais baixo que o superior, o que corresponde a quase cerca de dois gigabytes de diferença.

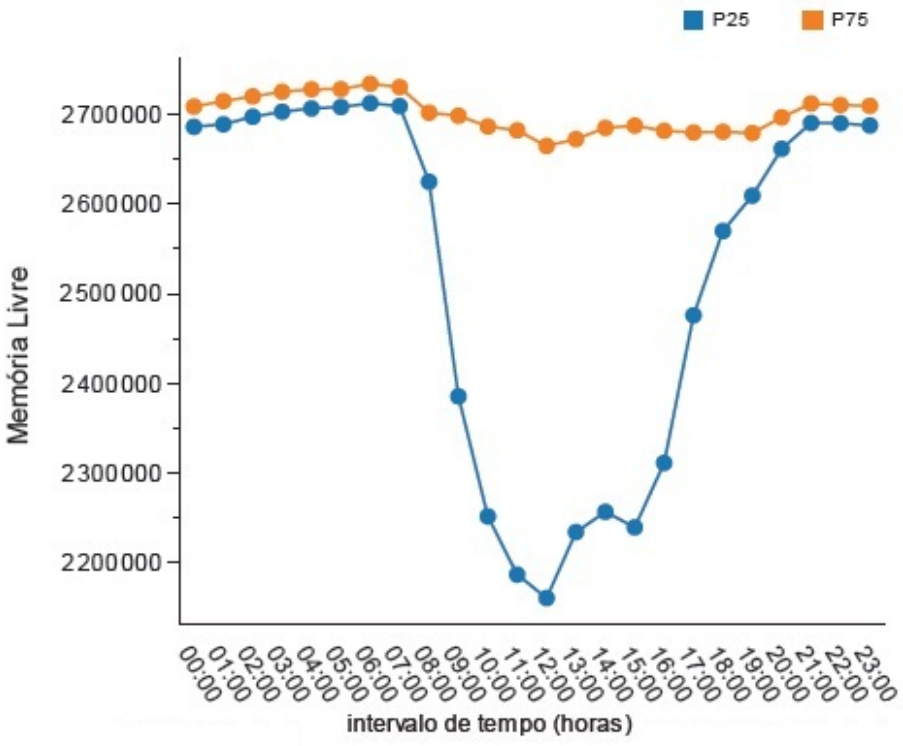


Figura 6.28: Memória livre ao longo do dia no SONHO

Rácio Buffer Cache

Através do gráfico da Figura 6.29, é possível observar desde logo que os valores dos limites do rácio buffer cache apresentam valores inferiores aos registados nos dois nodos da AIDA. Na maioria dos intervalos ao longo do dia, 50% dos dados encontram-se entre os 0,96 e os 0,94. Um facto curioso é que os melhores resultados deste rácio estão compreendidos entre as 8h00 e as 15h00 período onde normalmente existe uma maior carga na base de dados. Esta situação revela que a informação pretendida ao longo do dia não varia muito, ou então que a base de dados tem uma elevada capacidade de memória.

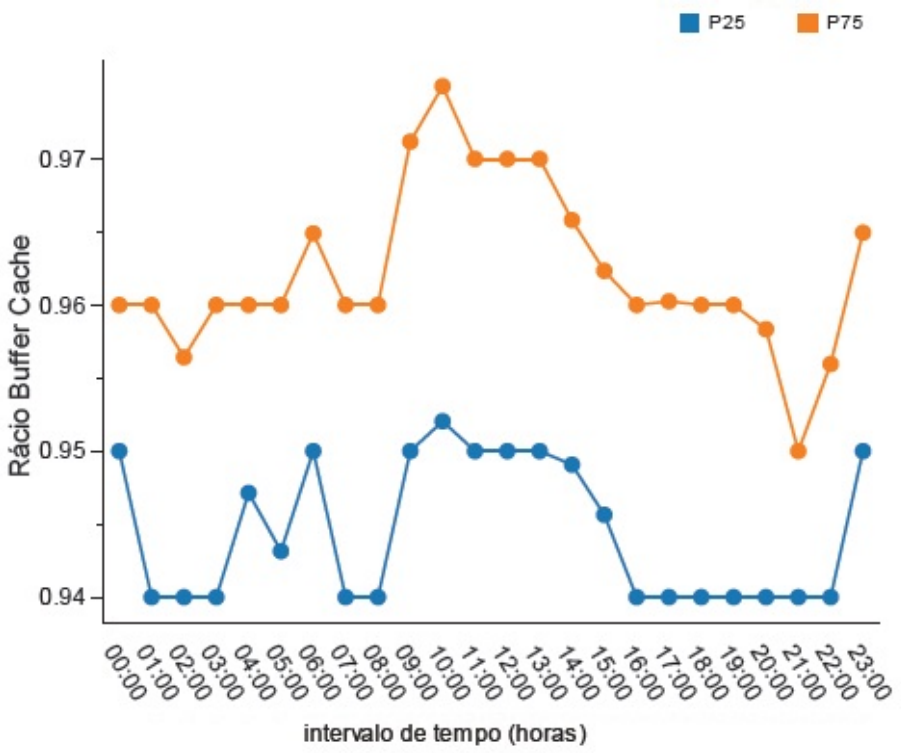


Figura 6.29: Limites rácio buffer cache ao longo do dia no SONHO

Rácio de chamadas recursivas

Relativamente ao rácio de chamadas recursivas este apresenta limites superiores elevados no início do dia que depois vão decrescendo. Como se pode observar na Figura 6.30, o valor mais elevado do percentil 75 é atingido no intervalo da 02h00, onde atinge quase os 0,9. No intervalos das 05h00 e 06h00 existe um outro pico de menor amplitude onde o percentil 75 ronda os 0,6. A partir das 08h00 começa a diminuir e mantêm-se abaixo dos 0,5 indicando que menos de metade das chamadas são chamadas recursivas, ou seja,

a maior parte das chamadas registadas são chamadas de utilizador.

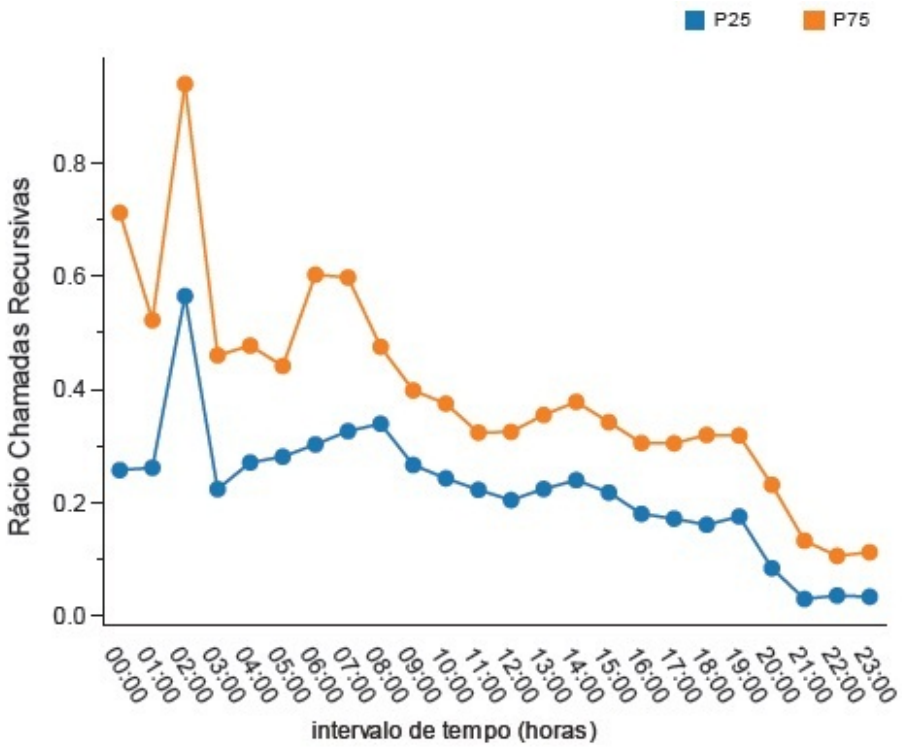


Figura 6.30: Limites do rácio de chamadas recursivas ao longo do dia no SONHO

Número operações de I/O

A Figura 6.31, representa os limites relativos aos pedidos de operações de I/O por segundo ao longo do dia. É notória a diferença entre o comportamento destes na parte noturna e na parte diurna. Durante a noite é possível verificar apenas um pico por volta das 02h00 onde 50% das medições situam-se entre os 50 e os 150 pedidos de operações I/O por segundo, estando este pico relacionado com operações de backup. Após isto o limite volta a um valor resídual abaixo de 10. Durante o dia, mais concretamente entre as 08h00 e as 18h00 o limite superior apresenta um comportamento em "forma de sino", sendo que o valor máximo atingido neste período é no intervalo das 10h00 onde o percentil 75 é de aproximadamente 70 pedidos de operações de I/O por segundo. Relativamente ao limite inferior, este apresenta sempre um valor resídual nunca ultrapassando os 25 pedidos por segundo durante o período diurno. É imporantante salientar que estes valores encontram-se muito a baixo dos valores dos limites para a mesma métrica na base de dados AIDA.

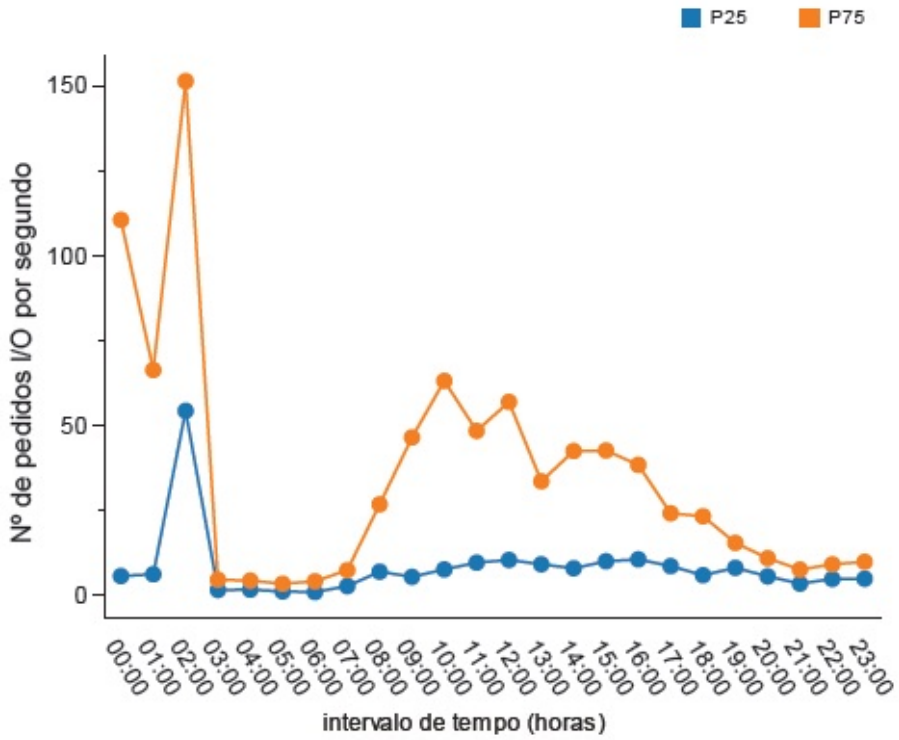


Figura 6.31: Limites do número de operações de I/O ao longo do dia no SONHO

Número de transações

Através da Figura 6.32, é possível observar o comportamento dos limites do número de transações por segundo ao longo do dia. O padrão desta distribuição é similar ao das métricas explicadas anteriormente, "em forma de sino". Excetuando o intervalo das 00h00 onde o valor do percentil 75 ultrapassa as 25 transações por segundo. No período compreendido entre as 08h00 e as 20h00 verifica-se então a curva em "forma de sino", onde o percentil 75 apresenta valores mais baixos no início e no final do período, apresentando valores mais elevados a meio do período. No intervalo das 12h00, o valor máximo atingido é aproximadamente igual ao verificado por volta das 00h00. O limite inferior não apresenta grandes oscilações variando ao longo do dia entre as 0 e 7 transações por segundo. Os valores desta métrica são também bastantes mais baixos para a base de dados SONHO do em cada um dos nodos da base de dados AIDA.

Número de Operações

A métrica operações por segundo apresenta valores muito elevados comparativamente com o que acontece com a AIDA, o que indica que apesar do baixo número de transações

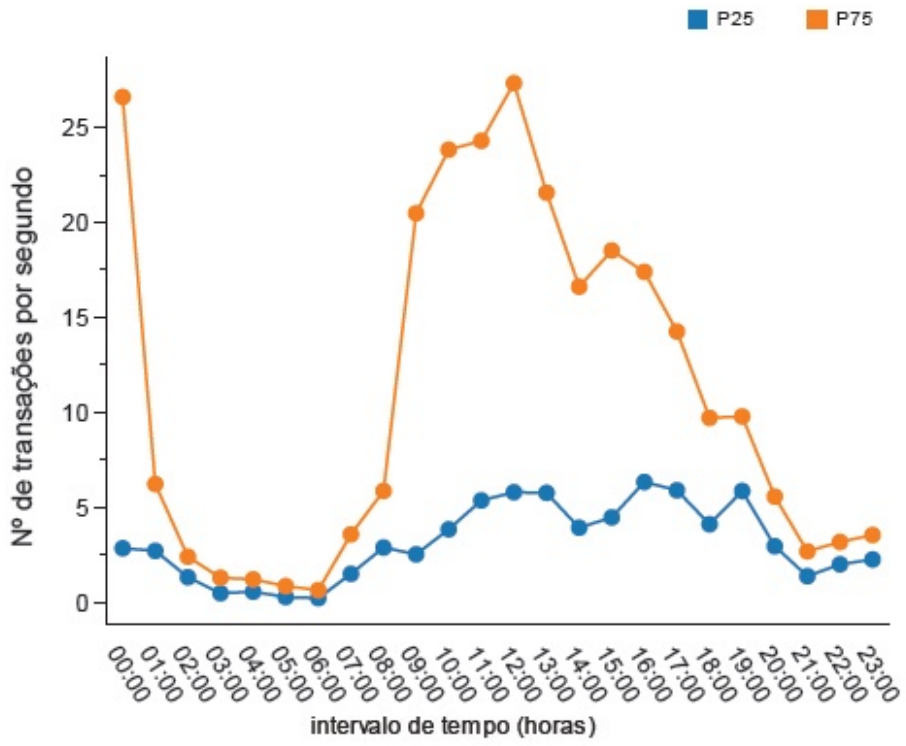


Figura 6.32: Limites do número de transações ao longo do dia no SONHO

no SONHO estas são bastante pesadas pois delas resulta um grande número de operações. Na Figura 6.33, é possível verificar que o pico máximo de operações por segundo é obtido no mesmo intervalo do máximo do rácio de chamadas recursivas, ou seja, no intervalo das 02h00, estando também relacionado com as rotinas de backup. Neste intervalo, o limite superior atinge valores muito altos de cerca de 8000 operações por segundo. Durante os restantes períodos do dia, os limites são substancialmente inferiores, sendo sempre inferiores às duas mil operações por segundo. No entanto, é possível verificar que no período compreendido entre as 09h00 e as 12h00 o valor do percentil 75 aproxima-se das 2000 operações por segundo. A partir das 13h00 o valor dos limites vai diminuindo até estabilizar por volta das 18h00.

Quantidade de entradas redo

Na Figura 6.34, apresenta-se o gráfico relativo aos limites do tamanho do ficheiro de *redo log*. Como é possível observar na Figura 6.34, o percentil 75 assume os valores máximos no período compreendido entre as 00h00 e as 02h00 ultrapassando o valor de 150 kb/s derivado às rotinas de backup que são efetuadas neste intervalo. Outro período que é

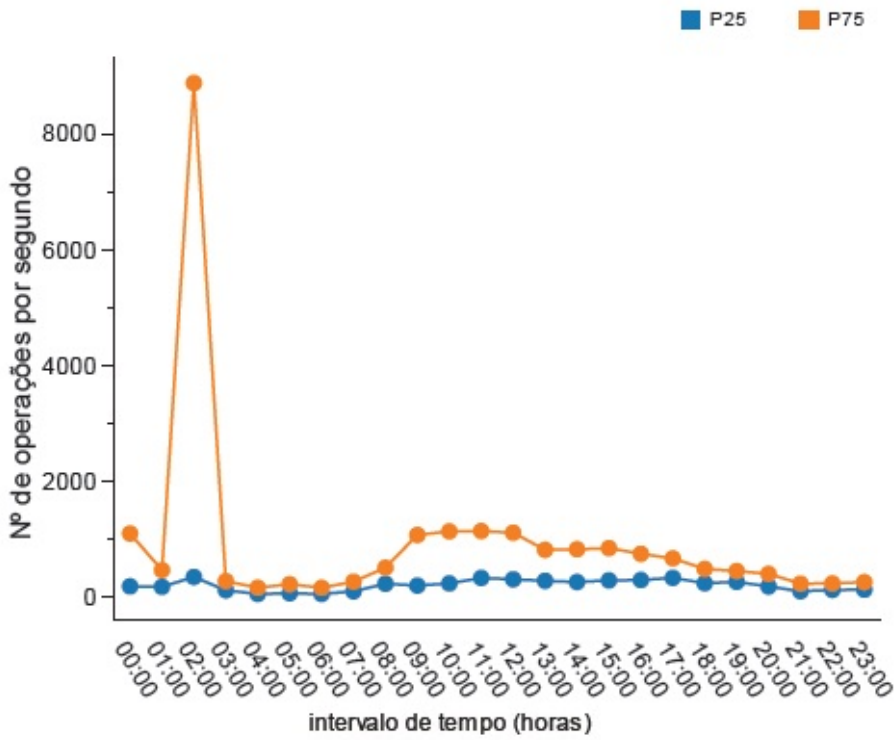


Figura 6.33: Limites do número de operações ao longo do dia no SONHO

importante salientar é o período compreendido entre as 09h00 e as 17h00. Neste período, o limite superior atinge valores na ordem dos 50 kb/s, sendo que no intervalo das 11h00 chega mesmo ultrapassar esta gama de valores. A partir deste intervalo o limite superior vai sofrendo uma ligeira diminuição até estabilizar rondando o valor de 11 kb/s.

Pedidos de espaço buffer redo log

Como já foi verificado no caso da base de dados AIDA, esta estatística normalmente apresenta valores muito pequenos. No caso da base de dados SONHO, como se pode verificar através do gráfico da Figura 6.35, estes valores são ainda mais pequenos. O limite inferior, representado pelo percentil 25 é zero ao londo de todo o dia, enquanto o percentil 75 sofre variações mas sempre na ordem das milésimas. Salientando-se os intervalos, das 00h00 e das 02h00 onde o valor atingido pelo percentil 75 foi um pouco superior a 0,006 pedidos por segundo. A meio do dia, por volta das 12h00 regista-se um novo aumento do percentil 75, mas desta vez apenas próximo dos 0,004 pedidos de espaço por segundo.

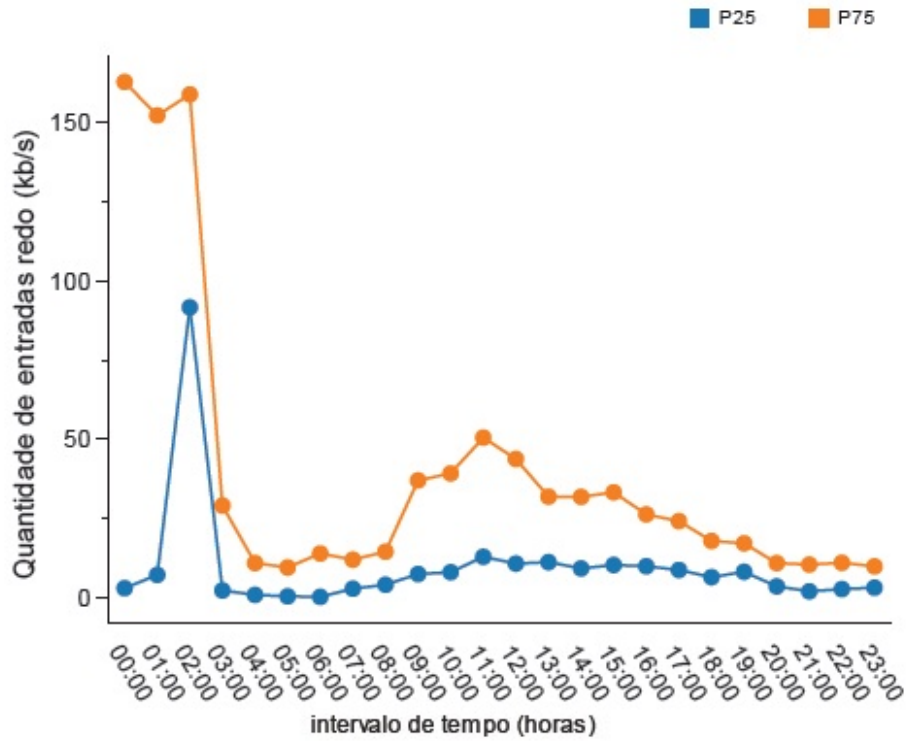


Figura 6.34: Limites da quantidade de entradas redo ao longo do dia no SONHO

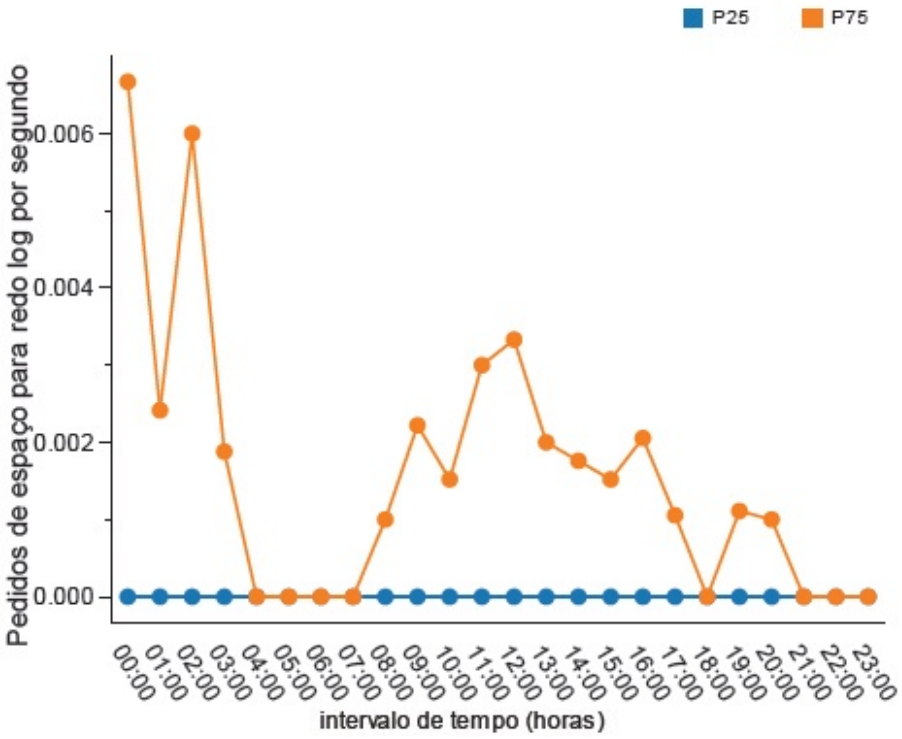
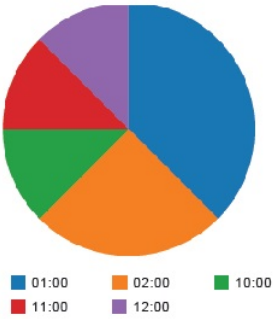


Figura 6.35: Limites do número de pedidos de espaço para o redo buffer ao longo do dia no SONHO

Avaliação dos picos

Para a base de dados SONHO foi elaborado um *dashboard* análogo ao aida_picos-wcdf, no entanto, o sonho_picos.wcdf apenas possui informação relativa à única instância do SONHO que fora avaliada. Na Figura 6.36, encontra-se um extrato deste *dashboard*. Através do gráfico da Figura 6.36, é possível observar as zonas onde se encontram os picos simultâneos das várias métricas, sendo que o intervalo que possuía mais picos na altura da medição era o intervalo das 01h00. Neste intervalo, o pico mais significativo é o relativo ao rácio de chamadas recursivas que é de 0,95, ou seja, muito próximo do máximo. É importante relembrar que este *dashboard*, de igual forma ao que acontece



Hora	Estatística	Valor
01:00	Quantidade de entradas redo (kb/s)	313.62
01:00	Rácio Chamadas Recursivas	0.95
01:00	Pedidos de espaço para redo log por segundo	0.02

Figura 6.36: Excerto dashboard sonho_picos.wcdf

com o da AIDA, é atualizado diariamente. Apesar de variar a quantidade de picos em cada intervalo, verifica-se que de dia para dia os intervalos dos picos mantêm-se, isto é, por vezes o pico de uma métrica está no intervalo das 10h00 enquanto no dia seguinte o pico dessa mesma métrica pode estar situado no intervalo adjacente, 11h00. Estas pequenas mudanças são frequentes. No entanto, verifica-se várias vezes que os períodos mais críticos se situam nos intervalos: 10h00, 11h00, 12h00, 01h00 e 02h00. Nestas alturas a base de dados está mais propícia a falhas.

6.2 Análise dos desvios à normalidade

Durante o período de execução do programa não se registaram falhas nas bases de dados em análise, não obstante, ocorreram vários desvios à normalidade, previamente estabelecida, fruto do dinamismo destas bases de dados. No entanto, o programa de monitorização foi re-inicializado algumas vezes, a maior parte delas automaticamente. O re-inicio

do programa ocorre quando existe a quebra de ligação à base de dados a monitorizar ou quando resulta algum erro da execução do programa, normalmente o programa re-inicia ficando registada a causa que levou à sua paragem. Em alguns casos foi necessário efetuar paragens rápidas e programadas para manutenção e correção de alguns erros relativos ao programa de monitorização.

Relativamente às anormalidades, como se pode observar na Tabela 6.3, a percentagem de situações anormais é inferior a 10% em cada uma das três bases de dados. O nodo 2

Tabela 6.3: Percentagem de anormalidades nas três bases de dados

Show 10 entries Search:

Base dados	Tot. med.	Tot. anormal	Pct_anormal
chp-ora01	164588	11975	7.28
chp-ora02	128757	10933	8.49
chp-sonho	142330	10720	7.53

da base de dados AIDA, o chp-ora02, é o que regista uma percentagem superior podendo isto dever-se ao facto da análise deste nodo ter começado mais tarde uma semana do que os restantes possuindo, desta forma, um número inferior de medições. Esta situação influencía os limites de normalidade previamente estabelecidos pois quanto maior o número de medições melhor é a capacidade do modelo representar a realidade.

Anormalidades distribuídas por score

Na Figura 6.37, encontra-se o gráfico que representa a distribuição de anormalidades por *score* para as três bases de dados. Como é possível observar, o *score* que regista um maior número de ocorrência, nas três bases de dados, é o *score* 3. Esta situação deve-se ao facto dos percentis utilizados para os cálculos dos limites serem relativamente próximos, muito facilmente, numa situação anormal, os limites indicativos dos *scores* 1 e 2 são ultrapassados. Através da Figura 6.37, é ainda possível observar que a distribuição de anormalidades nos nodos chp-ora01 e chp-ora02 é muito semelhante. Apenas se regista uma pequena diferença no *score* 3, onde o chp-ora02 apresenta um pouco mais ocorrências e no *score* 2 onde o número de ocorrências registadas é inferior no nodo chp-ora02. Relativamente ao SONHO, apresenta baixas ocorrências para o score 1 e 2, e um grande

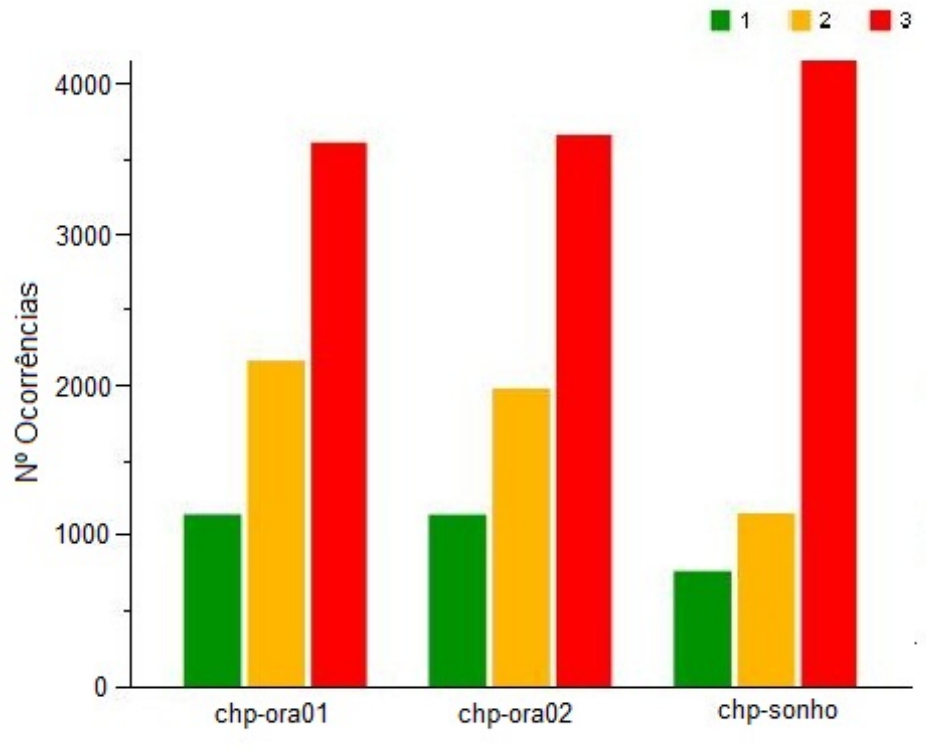


Figura 6.37: Número de situações anormais por score nas bases de dados do CHP

número de ocorrências no score 3 o que revela que os desvios à anormalidade nesta base de dados são mais intensos e preocupantes.

Anormalidades distribuídas por métrica

Outra análise interessante que se pode efetuar é avaliar a distribuição de anormalidades para cada métrica em cada base de dados. Apenas foram considerados os valores relativos a situações de score igual a 3, devido a representar as situações de maior gravidade. Na Figura 6.38, encontra-se um gráfico representativo desta situação. A intensidade das cores está de acordo com o número de ocorrências registadas, sendo que cores mais intensas corresponde a valores mais elevados. As métricas "DB time", e "trafego de rede", na base de dados SONHO estão marcadas com vermelho, pois como já foi visto anteriormente não foram recolhidas para esta base de dados. Como se pode observar na Figura 6.38, o rácio buffer cache é a métrica que apresenta um maior número de anormalidades nas três bases de dados. Este número elevado prende-se com a definição do número de casas decimais. Nas primeiras versões do programa, como este rácio variava pouco aumentou-se o número de casas decimais para ser possível observar pequenas variações. No entanto, verificou-

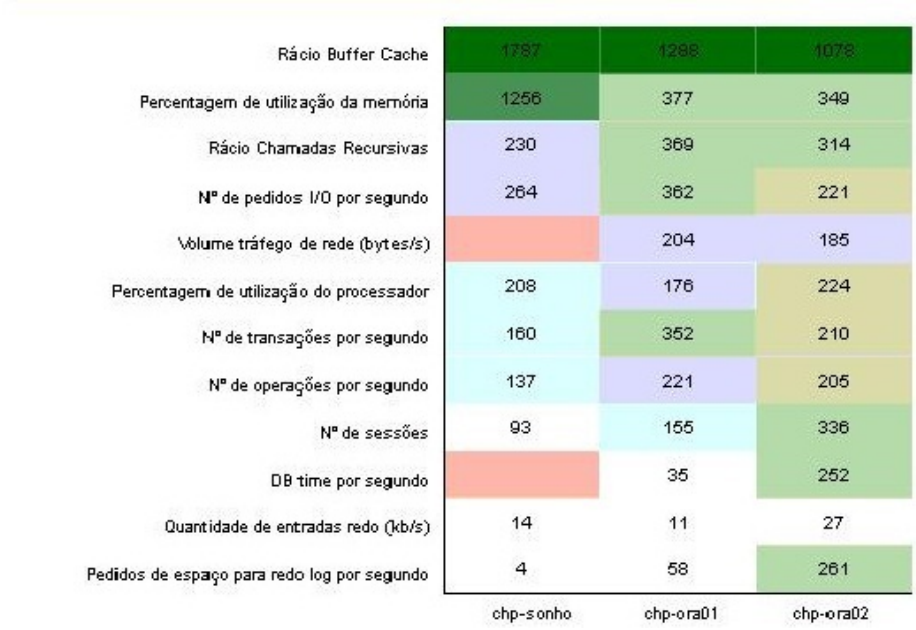


Figura 6.38: Número de situações anormais por métrica nas bases de dados do CHP

se que isso conduzia a um elevado número de "falsas anormalidades", então decidiu-se apertar de novo o número de casas decimais. Esta alteração já se encontra patente no novo programa de monitorização. A outra métrica que globalmente possui um maior número de ocorrências de anormalidades é a métrica relacionada com a utilização de memória. Enquanto que na AIDA os valores rondam as 350 ocorrências no SONHO esse valor é muito maior, perto das 1252 ocorrências.

A posição das restantes métricas varia de base de dados para base de dados, no entanto, é possível verificar também que nas três bases de dados as métricas relativas ao ficheiro *redo log* e aos pedidos de espaço para o *buffer redo* são as que apresentam um menor número de ocorrências de anormalidade, nunca ultrapassando as 60 ocorrências, excetuando o número de pedidos no nodo chp-ora02 que ultrapassa as 250 ocorrências.

Anormalidades por dia

Devido ao dinamismo das bases de dados do CHP, a contabilização do número de ocorrências anormais por dia é interessante para definir quais os períodos do mês em que elas mais ocorrem e se essa tendência é similar nas três bases de dados. Para tal, foram construídos os gráficos das Figuras 6.39, 6.40 e 6.41.

A Figura 6.39, corresponde à distribuição das anormalidades da base de dados AIDA. Como é possível de se observar na Figura 6.39, o dia em que se registou um maior número

de anormalidades foi o dia 31 de Julho, com o valor das anormalidades a rondar as 100 num só dia. Outro dia que se salienta é o dia 27 de Junho, onde o valor das situações anormais atingiu os 700. Nos restantes dias, o número de ocorrências é substancialmente homogéneo e encontra-se a baixo das 400 ocorrências por dia. Outra situação interessante é que durante os fins de semana, o valor das anormalidades normalmente é mais baixo que durante a semana, sendo tal facto explicado pela menor carga a que as bases de dados são sujeitas durante os fins de semana. Um exemplo é o fim de semana de 7 a 8 de Julho onde os valores não chegaram às 200 anormalidades por dia.

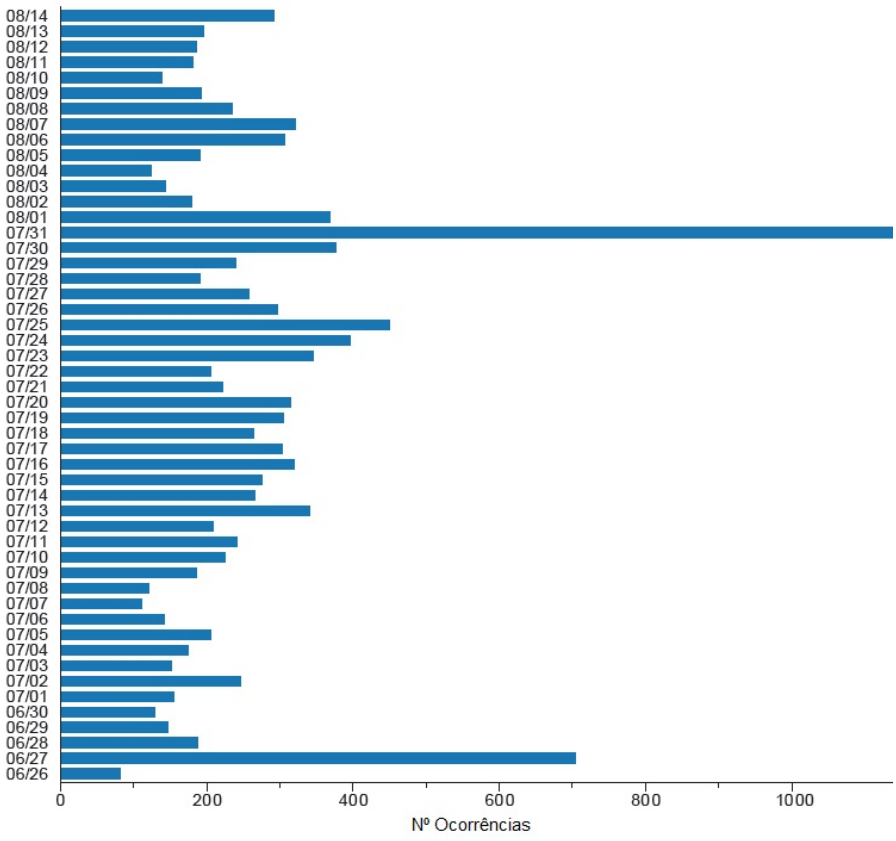


Figura 6.39: Número de situações anormais por dia (chp-ora01)

Os valores na Figura 6.40, relativa ao segundo nodo da AIDA (chp-ora02) são relativamente mais baixos. O dia em que se registou um maior número de ocorrências de anormalidades foi 5 de Julho onde o valor rondou as 650 anormalidades. Verifica-se também que após um mês de Julho mais calmo, no mês de Agosto tem existido um aumento de anormalidades por dia excetuando os fins de semana, onde os valores voltam a ser mais baixos. Por exemplo, no fim de semana de 4 e 5 de Agosto registou-se uma diminuição de anormalidades, curiosamente, na sexta-feira dia 3 verifica-se um valor mais baixo que

no domingo dia 5.

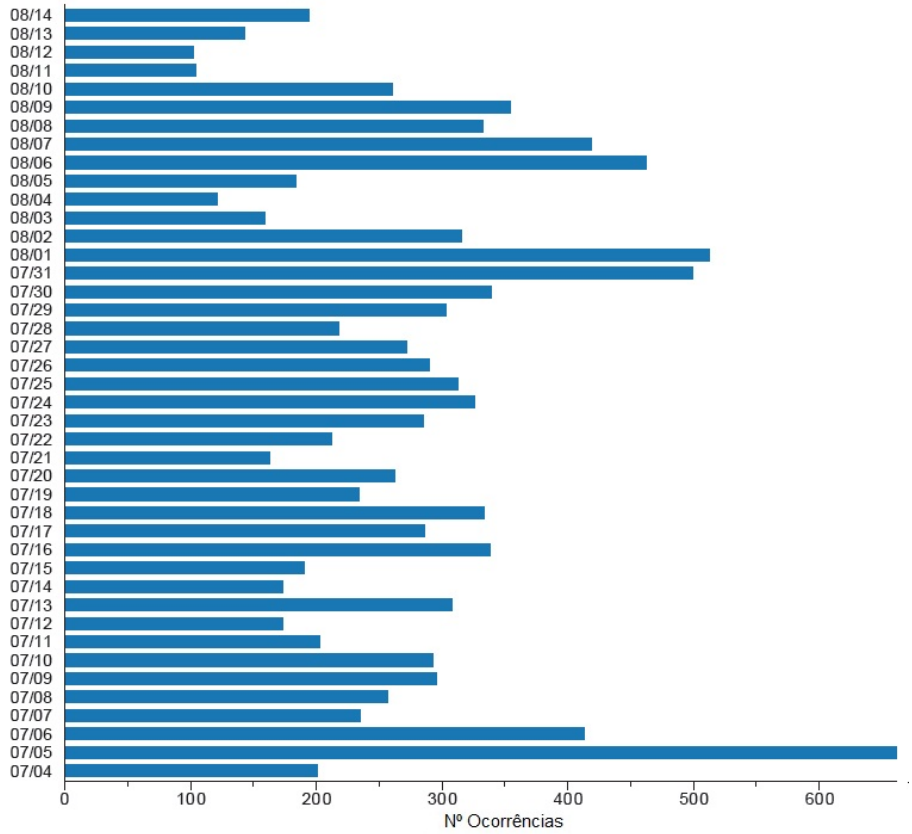


Figura 6.40: Número de situações anormais por dia (chp-ora02)

Relativamente à base de dados SONHO, pode-se observar na Figura 6.41, que esta apresenta um maior número de anormalidades em Agosto. O valor mais elevado de anormalidades, foi então registado no dia 8 de Agosto onde foram ultrapassadas as 400 ocorrências diárias. Apesar de em alguns fins de semana o número de anormalidades no SONHO ser mais baixo, esta situação não é tao clara como na AIDA existindo até fins de semana onde os valores atingidos estão muito próximos dos registados à semana. Tal como acontece no nodo 2 da AIDA, após um período mais calmo no mês de Julho nota-se um aumento significativo no número de anormalidades no início do mês de Agosto.

Exemplo situação anormal

Variadas situações anormais aconteceram durante o período delimitado para este estudo nas várias bases de dados. Decidiu-se apresentar um excerto do *dashboard* relativo à monitorização de uma das bases de dados para que fosse possível observar os indicadores visuais de alarme. A Figura 6.42 representa um excerto do *dashboard* de monitoriza-

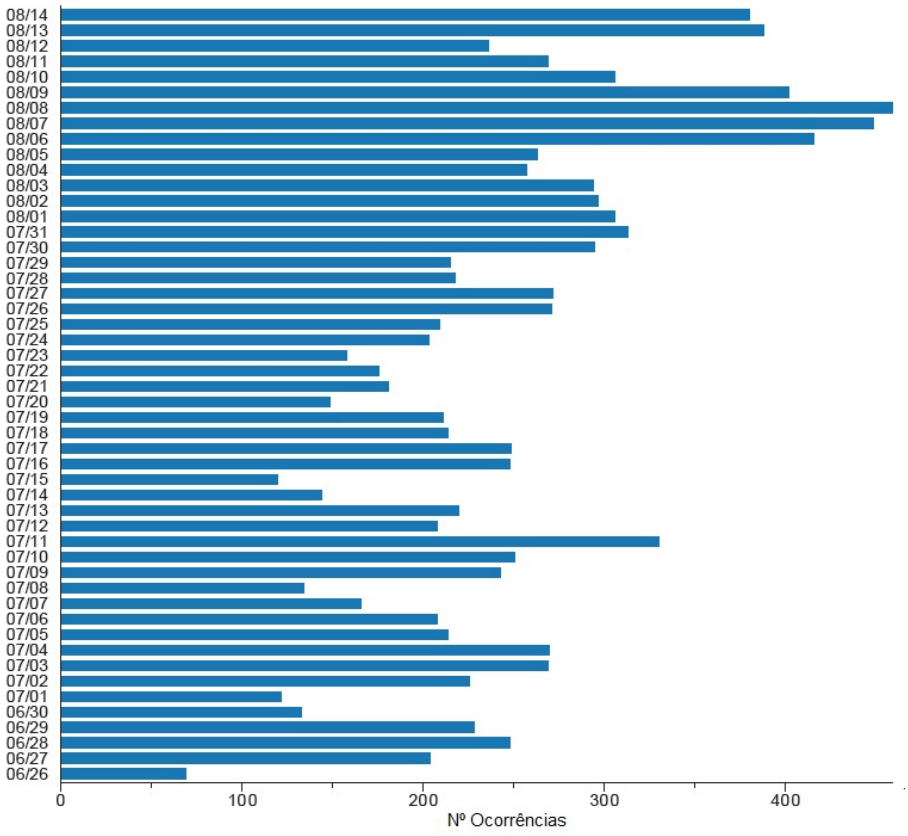


Figura 6.41: Número de situações anormais por dia (chp-sonho)

ção do nodo chp-ora01 da AIDA, nela estão contidos seis gráficos relativos às métricas: número de sessões, percentagem de memória, volume do tráfego de rede, número de transações, operações e pedidos de I/O por segundo. Em todos estes gráficos encontram-se quatro linhas, a verde corresponde ao percentil 75, a amarela ao percentil 80, a vermelha ao percentil 90 e a azul corresponde ao valor medido a cada minuto. Este excerto foi retirado num dos períodos críticos do dia, verificando-se que a linha azul ultrapassa com alguma frequência os limites estabelecidos pelos percentis indicando a presença de uma situação anormal.

Se apenas for considerada a última medição verifica-se que o número de sessões e o volume do trafego de rede encontram-se acima do percentil 90, o número de operações por segundo está acima do percentil 80, e o número de pedidos de operações I/O está entre o percentil 75 e 80. As restantes métricas encontram-se numa situação normal, ou seja, abaixo do percentil 75.

Neste caso, a soma do score global seria 9, o que originaria um alerta de anormalidade via email. No entanto, é importante salientar, que os emails apenas são enviados de 15 em

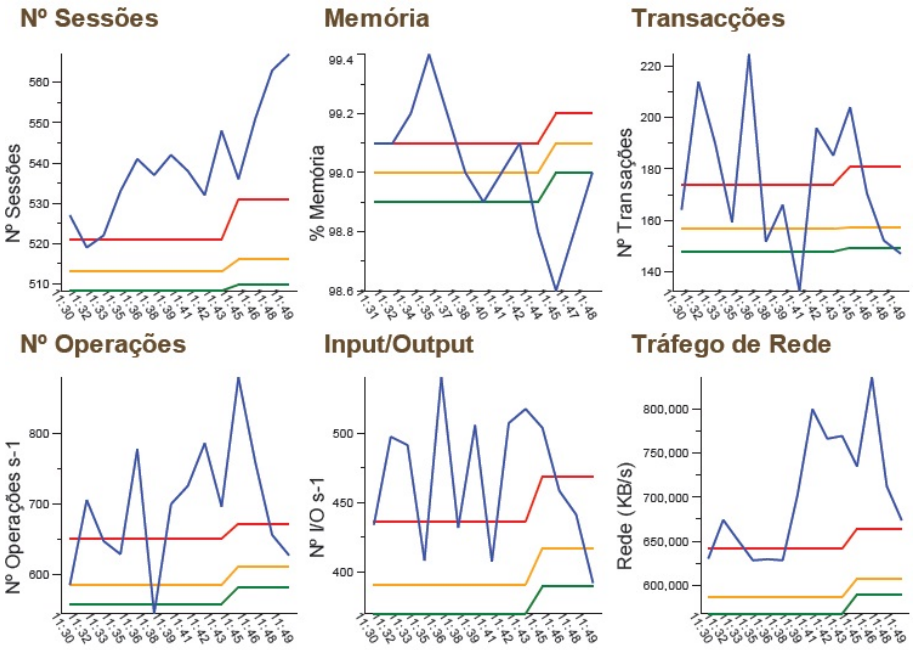


Figura 6.42: Exemplo de uma situação anormal (excerto do dashboard chp-ora01)

15 minutos para que possam ser utilizados valores médios para a comparação evitando desta forma o impacto das pequenas oscilações durante o intervalo.

Capítulo 7

Publicações

A execução desta dissertação originou três publicações, sendo estas na forma de artigos para conferências internacionais, nomeadamente anexo F:

Título: Step Towards Fault Forecasting in Hospital Information Systems [64]

Local: 5th International Conference on BioMedical Engineering and Informatics (Chongqing, China)

Resumo: Hoje em dia, muitas organizações consideram as bases de dados como ferramentas indispensáveis para as suas tarefas do dia-a-dia. No caso particular das unidades de saúde, as bases de dados assumem um papel vital, pois elas armazenam informações muito importantes sobre o estado clínico do paciente. Por esta razão, é crucial que as bases de dados estejam disponíveis vinte e quatro horas por dia, sete dias por semana. As unidades de saúde já possuem implementados mecanismos de tolerância a falhas, que tentam garantir a disponibilidade, confiabilidade e a recuperação de falhas dos dados. No entanto, estes mecanismos não permitem a tomada de ações preventivas para evitar falhas. Neste contexto, emerge a necessidade de desenvolver sistemas de prevenção e previsão de falhas. Estes sistemas podem prever, com algum tempo de antecedência, as falhas possibilitando a tomada de ações para tentar evitar que as falhas ocorram. Os objetivos deste artigo são: monitoriza a performance da base de dados, adaptar o modelo usado na medicina (Modified Early Warning Score) à realidade das bases de dados hospitalares.

Factos Relevantes: Neste artigo são apresentadas as arquiteturas das bases de dados em estudo: AIDA e SONHO. Para além disso, é descrito o modelo MEWS, as métricas usadas para o programa de monitorização e é apresentado o programa de BI usado: Pentaho Community. É também apresentada a ideia geral da representação do comportamento

normal das bases de dados com a ajuda de ferramentas matemáticas, nomeadamente o uso de percentis para o cálculo dos limites.

Principais Conclusões: Apesar do tema de previsão de falhas ser bastante complexo, com o modelo apresentado no artigo é possível representar o normal funcionamento das bases de dados apresentadas. De acordo com a frequência e a intensidade das situações anormais é possível criar uma tabela de decisão que representem a gravidade da situação.

Título: Intelligent Systems based in Hospital Database Malfunction Scenarios [65]

Local: The IEEE International Conference on Industrial Engineering and Engineering Management (Hong Kong, China)

Resumo: As bases de dados são indispensáveis para as tarefas de todos os dias em muitas organizações, particularmente nas organizações de saúde. As bases de dados, entre outras operações relevantes, permitem o armazenamento de informação importante e privada sobre o estado clínico do paciente. Desta forma, elas têm de estar disponíveis, seguras e a um alto nível de performance durante as vinte e quatro horas do dia, e os sete dias da semana. Em muitas organizações de saúde, já se encontram em funcionamento sistemas de tolerância a falhas que garantem a disponibilidade, segurança e capacidade de recuperação dos dados. No entanto, estes mecanismos não permitem tomar ações preventivas para evitar as falhas. Neste contexto, é de elevada importância, a necessidade de desenvolver um sistema de prevenção de falhas. A intenção deste artigo é promover a monitorização da performance das bases de dados e adaptar o modelo usado na medicina (MEWS) ao contexto das bases de dados. Com base em ferramentas matemáticas foi criada uma escala que representa o grau de gravidade das situações anormais detetadas. Desta forma, é possível a determinação de senários onde os sintomas das bases de dados desencadeiam alertas e pedidos de assistência.

Factos Relevantes: Neste artigo são apresentados todos os elementos que permitem perceber o funcionamento do modelo (arquitetura das Bases de Dados, descrição do MEWS, apresentação do Pentaho, descrição das métricas usadas, metodologia seguida). Para além disso, é apresentada a tabela que representa a escala de gravidade das situações anormais. Os níveis desta escala têm por base os percentis dos dados recolhidos, ou seja, em função dos dados recolhidos posteriormente é possível definir os limites representativos do normal comportamento de cada métrica.

Principais Conclusões: Com este modelo de representação de carga de trabalho é possível identificar picos de utilização para cada métrica e encontrar os períodos do dia nos quais a probabilidade de ocorrência de falhas é maior. Através da tabela de decisão é possível atribuir *scores* em função dos desvios dos limites previamente definidos para cada métrica. Desta forma, o MEWS foi adaptado com sucesso através de um programa atualizável. O Pentaho Community revelou-se útil para a manipulação dos dados e para o desenvolvimento de *dashboards* que facilitam a interpretação de resultados.

Título: Hospital database workload and fault forecasting [66]

Local: International Conference on Biomedical Engineering and Sciences (Langkawi, Malásia)

Resumo: Com a crescente importância dos sistemas de informação hospitalar, as bases de dados tornaram-se ferramentas indispensáveis para as tarefas do dia-a-dia nas unidades hospitalares. Elas armazenam informações importantes e confidenciais sobre o estado clínico do paciente e sobre os outros serviços hospitalares. Desta forma, elas têm de estar permanentemente disponíveis, seguras e num nível de performance elevado. Elas garantem a disponibilidade e recuperação dos dados. Contudo, estes mecanismos não permitem nem a prevenção nem a previsão de falhas. Neste contexto, emerge a necessidade de desenvolver um sistema de previsão de falhas. Os objetivos deste artigo são monitorizar a performance da principal base de dados do Centro Hospitalar do Porto e adaptar o modelo de previsão usado na medicina ao caso das bases de dados. Com base nos percentis foi elaborada uma escala que representa o grau de gravidade das situações anormais. Observou-se que o período crítico a nível de carga de trabalho situa-se entre as 10:00 e as 12:00. Para além disso, situações anormais foram detetadas e foi possível o envio de alertas para pedir assistência.

Factos Relevantes: Neste artigo foram descritos os elementos essenciais para a implementação do modelo (descrição do MEWS, Pentaho, métricas usadas e funcionamento do modelo), no entanto, foi abordado com mais pormenor as características relativas à base de dados AIDA e ao sistema PCE. Devido à base de dados AIDA possuir uma arquitetura RAC, foi elaborado um estudo aos dois nodos para perceber de que forma a carga estava distribuída entre eles. Além disso, foi também feita a análise global à performance da base de dados AIDA. Após a determinação do comportamento normal das bases de

dados foram definidos os limites e com base nos percentis procedeu-se à recolha de anormalidades. Dependendo da gravidade da situação são mostrados indicadores visuais ou então é enviado um email alertando para essa situação.

Principais Conclusões: Através do modelo proposto para caraterizar a carga de trabalho da base de dados AIDA observou-se que o nodo 1 apresenta maior carga que o nodo 2. Além disso, foi possível verificar que a maioria das métricas apresenta picos no intervalo das 10:00 às 12h00. Foi elaborada uma tabela de decisão que com base nos percentis adapta o MEWS às base de dados. Devido ao dinamismo da base de dados os limites têm de ser atualizados todos os dias. Concluí-se também que o Pentaho Community é uma ferramenta bastante completa que permite a criação de *dashboads* o que facilita o processo de interpretação dos dados.

Está ainda em elaboração um artigo para submeter a uma revista, assim como outro artigo que está pronto para ser submetido numa conferência internacional indexada nos principais sistemas científicos.

Capítulo 8

Conclusões

8.1 Visão geral

No CHP as bases de dados são ferramentas fundamentais para a qualidade dos serviços prestados contendo as informações necessárias para todos os sistemas de informação. Apesar destas bases de dados estarem bem aprovisionadas no que respeita à tolerância a falhas, verificou-se que devido à sua elevada carga e utilização, estas necessitavam de um sistema de alarmística que possibilitasse a previsão e prevenção de falhas.

Apesar da base de dados SONHO ser distinta da base de dados AIDA, constatou-se que possuem em comum um elevado número médio de utilizadores, um elevado número médio de operações, bem como um elevado consumo de memória. Para além disso, constatou-se que o desvio padrão era elevado na maior parte das métricas o que representa a grade variabilidade dos resultados recolhidos. Esta situação deve-se ao dinamismo das bases de dados analisadas e ao facto de serem frequentemente adicionados novos módulos aos sistemas de informação que usam as bases de dados o que leva ao incremento da carga das mesmas.

No caso particular da base de dados AIDA, devido a possuir uma arquitetura RAC realizou-se a análise aos dois nodos em separado. Através desta análise foi possível observar que o nodo 1 (chp-ora01) apresenta valores médios ligeiramente mais elevados do que o nodo 2 (chp-ora02) em quase todas as métricas excetuando as métricas: percentagem de uso de processador e quantidade de entradas *redo log*.

Normalmente a distribuição dos limites nos dois nodos é bastante similar para quase todas as métricas, diferenciando-se apenas na amplitude dos valores. No entanto, verificou-

se que relativamente à métrica de percentagem de uso de memória esta apresenta diferenças significativas entre os dois nodos, sendo que no nodo 2 apresenta uma elevada percentagem durante a madrugada o que não acontece no nodo 1. Este valor elevado de memória durante a noite está relacionado com alguns processos mais pesados que são executados neste período aproveitando uma menor afluência à base de dados.

Constatou-se que as diferentes métricas da base de dados AIDA podem ser agrupadas de acordo com a tendência dos limites ao longo do dia. Sendo assim é possível separá-las em três grupos. No primeiro grupo encontram-se aquelas que apresentam uma curva em "forma de sino", ou seja, valores elevados a meio do dia e mais baixos durante o fim de tarde, à noite e no início da manhã. Deste grupo fazem parte as métricas: número de sessões, percentagem de uso de processador, percentagem de uso de memória (nodo 1), volume do tráfego da rede e o db time.

O segundo grupo é formado pelas métricas que apresentam alguns picos durantes a noite, mais concretamente nos intervalos 22h00, 01h00, 05h00. Deste grupo fazem parte os rácios buffer cache e chamadas recursivas, e também os pedidos de operações I/O e o número de operações por segundo.

O terceiro grupo é composto por métricas com comportamentos distintos dos outros dois grupos mencionados acima. As métricas quantidade de entradas redo e os pedidos de espaço para o buffer redo possuem a mesma tendência, apresentando um pico por volta da 01h00 e mantendo-se constante ao longo do dia. Por sua vez, o número de transações por segundo apresenta uma grande variabilidade e uma tendência diferente de todas as métricas aqui apresentadas.

As métricas do segundo e terceiro grupo, excetuando o número de transações, são as métricas mais influenciáveis pelas rotinas de backup.

Globalmente é importante salientar que nos intervalos das 11h00 e 12h00, bem como nos intervalos das 00h00 e da 01h00 encontra-se a maior parte dos picos das métricas de ambos os nodos da base de dados AIDA. Por esta razão pode-se considerar estes períodos como os mais críticos nos quais a atenção deve ser redobrada de forma a evitar possíveis falhas.

Relativamente à base de dados SONHO após ser efetuada uma análise similar à da AIDA. Constatou-se que a maioria das métricas (número de sessões, percentagem de uso de processador, memória, número de pedidos I/O, número de transações por segundo,

número de pedidos de espaço) apresenta valores mais elevados durante o período das 08h00 às 17h00, apresentando uma curva em "forma de sino" durante este período. No entanto, verifica-se que algumas delas apresentam alguns picos em períodos distintos do dia, como é o caso do rácio de chamadas recursivas, do número de operações por segundo e da quantidade de entradas redo que apresentam picos às 02h00; o número de transações por segundo e a quantidade de pedidos de espaço para *buffer redo* que têm picos às 00h00. O buffer cache possui um comportamento curioso apresentando melhores valores durante o período mais crítico demonstrando um bom dimensionamento da memória.

Apesar dos picos variarem ao longo dos dias, como acontece com os relativos às bases de dados AIDA, os intervalos onde se registam mais vezes picos na base de dados SONHO são: 10h00, 11h00, 12h00, 00h00 e 02h00.

Após algumas experiências foi construída a tabela de decisão cujo objetivo é implementar uma escala de gravidade assim como acontece no MEWS. A escala foi elaborada tendo em conta os percentis 75, 80 e 90. Verificou-se que estes limites deviam ser atualizados diariamente para que melhor representem a realidade das duas bases de dados, pois quanto maior o número de medições maior é a qualidade do modelo. ´

Durante o período de execução do programa final, não ocorreram quaisquer falhas nas bases de dados. Apurou-se que a percentagem de anormalidades detetadas é relativamente baixa e com tendência a baixar ainda mais o que revela que aos poucos o modelo vai representando melhor a realidade. Pela mesma razão a instância de base de dados que revela um maior número de anormalidades é o nodo chp-ora02, onde o programa de monitorização entrou em funcionamento uma semana mais tarde.

Apurou-se também que das anormalidades detetadas a maioria delas são anormalidades de score 3 (score mais grave) devendo-se isto ao facto dos limites serem relativamente próximos. Foi ainda possível identificar quais as métricas onde são registadas mais anormalidades do score 3, por serem as situações mais gravosas. Nas três bases de dados o buffer cache foi a métrica que obteve mais anormalidades fruto do número de casas decimais usadas aquando a sua recolha, que fora entretanto ajustado. A outra métrica que nas três bases de dados originou mais anormalidades foi a percentagem de utilização de memória. No lado oposto, a métrica que originou menos anormalidades foi a quantidade de entradas *redo log*.

Verificou-se ainda que, no caso da base de dados AIDA durante os fins-de-semana

ocorrem menos anormalidades, fruto da diminuição da carga de trabalho. O mesmo, curiosamente, não acontece na base de dados SONHO. Nesta base de dados apenas em alguns domingos o nível de anormalidades é menor, de resto não se pode tirar mais nenhuma elação.

Apesar de ainda existirem aspetos a melhorar, conclui-se que a adaptação do mecanismo MEWS à realidade das bases de dados é possível e tem bons resultados no que toca à alarmística de situações anormais. Acredita-se vivamente que, efetuadas algumas melhorias, este modelo tem capacidade de se tornar numa ferramenta poderosa de auxílio aos administradores das bases de dados hospitalares e consequentemente melhorar a qualidade dos serviços prestadas por estas.

8.2 Trabalho futuro

Em termos de trabalho futuro seria importante verificar o comportamento do modelo passados mais alguns meses, pois dinamicamente o modelo vai sendo atualizado o que leva a que um maior período de execução conduza a melhores resultados.

Seria uma mais-valia também, a simulação de algumas situações que originem falhas. Este estudo iria permitir a criação de um limite de falha iminente, individual para cada métrica, ao qual seria atribuído o score 3. Esta situação é um pouco difícil de implementar devido à importância destas bases de dados, um pequeno período de indisponibilidade poderia trazer muitos problemas à prestação dos serviços de saúde pelo CHP.

Outro aspeto a melhorar seria a inclusão de mais métricas, para que, após a simulação de falhas fosse possível identificar quais as que revelam mais impacto. Desta situação resultariam substanciais mais-valias pois as métricas seriam escolhidas para um caso concreto e não baseadas na literatura.

Bibliografia

[1] C. P. Subbe, M. Kruger, P. Rutherford, and L. Gemmel, "Validation of a Modified Early Warning Score in medical admissions," *QJM*, vol. 94, no. 10, pp. 521–526, 2001.

[2] T. Connolly and C. Begg, *Database Systems A Pratical Approach to Design, Implementation, and Management*, 4th ed. Harlow: Addison-Wesley, 2005.

[3] A. Rodrigues, *Oracle 10g e 9i: Fundamentos Para Profissionais*. Lisboa: FCA, 2005.

[4] J. A. Hoffer, M. B. Prescott, and F. R. McFadden, *Modern Database Management*, 8th ed. Upper Saddle River: Prentice Hall, 2007.

[5] S. Paul, "Database Systems Performance Evaluation Techniques," 2008.

[6] G. W. Hansen and J. V. Hansen, *Database Management and Design*, 2nd ed. Upper Saddle River, N.J.: Prentice Hall, 1996.

[7] A. J. Oppel, *Databases demystified - a self-teaching guide*. New York: McGraw-Hill Osborne, 2004.

[8] M. Cyran, P. Lane, and J. Polk, *Oracle Database Concepts, 10g Release 2 (10.2)*. Oracle, 2005.

[9] E. Bertino and R. Sandhu, "Database security - concepts, approaches, and challenges," *Dependable and Secure Computing, IEEE Transactions on*, vol. 2, no. 1, pp. 2–19, 2005.

[10] S. Kim, N. Cho, Y. Lee, S.-H. Kang, T. Kim, H. Hwang, and D. Mun, "Application of density-based outlier detection to database activity monitoring," *Information Systems Frontiers*, pp. 1–11, 2010.

[11] R. Godinho, "Availability, Reliability and Scalability in Database Architecture," Mestrado em Informática, Universidade do Minho, 2011.

[12] O. Yi, "Highly Available Database Systems," in *Integrated information systems and databases seminar*, 2006.

[13] C. Shallahamer, *Forecasting Oracle Performance*. Berkeley: Apress, 2007.

[14] F. Piedad, *High availability : design, techniques, and processes*. Upper Saddle River: Prentice Hall, 2001.

[15] A. Babb, T. Bednar, L. Carpenter, I. Chan, R. Dutcher, C. Foch, F. Kobylanski, B. Lundhild, R. Mau, J. Meeks, V. Moore, M. Nowak, A. Ray, V. Schupmann, M. Smith, L. To, D. Utzig, J. Viscusi, and S. Yamaguchi, *Oracle Database High Availability Overview, 11g Release 1 (11.1)*. Oracle, 2007.

[16] M. Bauer and R. Strohm, *Oracle Real Application Clusters Administration and Deployment Guide, 11g Release 1 (11.1)*. Oracle, 2009.

[17] S. Drake, W. Hu, D. McInnis, M. Sköld, A. Srivastava, L. Thalmann, M. Tikkanen, O. y. Torbjø rnsen, and A. Wolski, "Architecture of Highly Available Databases," in *Service Availability*, ser. Lecture Notes in Computer Science, M. Malek, M. Reitenspieß, and J. Kaiser, Eds. Springer Berlin / Heidelberg, 2005, vol. 3335, pp. 1–16.

[18] M. Wiesmann, F. Pedone, A. Schiper, B. Kemme, and G. Alonso, "Understanding Replication in Databases and Distributed Systems," 2000, pp. 264–274.

[19] D. Shasha and P. Bonnet, *Database tuning : principles, experiments, and troubleshooting techniques*. Amsterdam: Morgan Kaufmann Publishers, 2003.

[20] T. Hall, "ORACLE-BASE - Oracle And RAID," acedido a: 20/01/2012. [Online]. Disponível em: http://www.oracle-base.com/articles/misc/RAID.php#OracleRAIDUsage

[21] W. Hodak, S. Kumar, and A. Ray, "Oracle Database 11g High Availability," Oracle, Tech. Rep., 2007.

[22] C. F. Portela, M. F. Santos, A. Silva, J. Machado, and A. Abelha, "Enabling a pervasive approach for intelligent decision support in critical health care," in *ENTERprise Information Systems*, ser. Communications in Computer and Information Science, M. M. Cruz-Cunha, J. VarajÃ£o, P. Powell, and R. Martinho, Eds. Springer Berlin Heidelberg, 2011, vol. 221, pp. 233–243.

[23] A. Albino and V. Jacinto, "Implementação da escala de alerta precoce - EWS," Centro Hospitalar do Barlavento Algarvio, EPE, Portimão, Tech. Rep., 2009.

[24] F. Portela, P. Gago, M. F. Santos, A. Silva, F. Rua, J. Machado, A. Abelha, and J. Neves, "Knowledge discovery for pervasive and real-time intelligent decision support in intensive care medicine," in *KMIS 2011- International Conference on Knowledge Management and Information Sharing*, Paris, 2011.

[25] MedlinePlus, "Medical Dictionary," acedido a: 6/01/2012. [Online]. Disponível em: http://www.merriam-webster.com/medlineplus/

[26] G. Devaney and W. Lead, "Guideline for the use of the modified early warning score (MEWS)," Outer North East London Community Services, Tech. Rep., 2011.

[27] J. Gardner-Thorpe, N. Love, J. Wrightson, S. Walsh, and N. Keeling, "The value of Modified Early Warning Score (MEWS) in surgical in-patients: a prospective observational study." *Annals of the Royal College of Surgeons of England*, vol. 88, no. 6, pp. 571–5, Oct. 2006.

[28] C. P. Subbe, R. G. Davies, E. Williams, P. Rutherford, and L. Gemmell, "Effect of introducing the Modified Early Warning score on clinical outcomes, cardio-pulmonary arrests and intensive care utilisation in acute medical admissions*," *Anaesthesia*, vol. 58, no. 8, pp. 797–802, 2003.

[29] M. Santos and F. Portela, "Enabling ubiquitous data mining in intensive care - features selection and data pre-processing," in *ICEIS (1)'11*, 2011, pp. 261–266.

[30] N. N. Savino-Vázquez, J. L. Anciano-Martin, S. Dumas, J. A. Corbacho, R. Puigjaner, D. Boudigue, and G. Gardarin, "Predicting the behaviour of three-tiered applications: dealing with distributed-object technology and databases," *Performance Evaluation*, vol. 39, no. 1-4, pp. 207–233, Feb. 2000.

[31] J. Duggan, U. Cetintemel, O. Papaemmanouil, and E. Upfal, "Performance prediction for concurrent database workloads," in *Proceedings of the 2011 international conference on Management of data*, ser. SIGMOD '11. New York, NY, USA: ACM, 2011, pp. 337–348.

[32] F. Salfner, M. Lenk, and M. Malek, "A survey of online failure prediction methods," *ACM Comput. Surv.*, vol. 42, no. 3, pp. 10:1–10:42, Mar. 2010.

[33] M. P. Sullivan, "System Support for Software Fault Tolerance in Highly Available Database Management Systems," Tech. Rep., 1992.

[34] S. Nair, "The Art of Database Monitoring," *Information Systems Control Journal*, no. Ccm, pp. 1–4, 2008.

[35] I. Chan, *Oracle Database Performance Tuning Guide, 10g Release 2 (10.2)*. Oracle, 2008.

[36] X. Hong, "A Database Monitoring and Disaster Recovery System," in *Frontiers of WWW Research and Development - APWeb 2006*, ser. Lecture Notes in Computer Science, X. Zhou, J. Li, H. Shen, M. Kitsuregawa, and Y. Zhang, Eds. Springer Berlin / Heidelberg, 2006, vol. 3841, pp. 1201–1204.

[37] P. Rob and C. Coronel, *Database systems : design, implementation, and management*. Boston: Thomson, 2004.

[38] L. Ramos, "Performance Analysis of a Database Caching System In a Grid Environment," Mestrado em Engenharia Informática, FEUP, 2007.

[39] R. Schumacher, *Oracle Performance Troubleshooting With Dictionary Internals SQL & Tuning Scripts*. Kittrell: Rampant TechPress, 2003.

[40] Database Specialists, "A Comparison of Proactive Monitoring Methods for Oracle Databases," 2009.

[41] I. Chan, *Oracle Database 2 Day + Performance Tuning Guide, 10g Release 2 (10.2)*. Oracle, 2010.

[42] Nagios, "Nagios - Nagios Core," acedido a: 4/10/2011. [Online]. Disponível em: http://www.nagios.org/projects/nagioscore

[43] I. Jones and R. Schrag, "Finding the Performance Bottlenecks in Your Application," 1999.

[44] K. Rich, *Oracle Database Reference, 10g Release 2 (10.2)*. Oracle, 2009.

[45] K. Dias, M. Ramacher, U. Shaft, V. Venkataramani, and G. Wood, "Automatic performance diagnosis and tuning in Oracle," in *Proceedings of the 2005 CIDR Conf*, 2005.

[46] DeniX Solutions SRL, "Linux, FreeBSD, OpenBSD, NetBSD, HP-UX, Tru64 Unix Documentation Project," 2005, acedido a: 20/02/2012. [Online]. Disponível em: http://nixdoc.net/

[47] J. Beresniewicz, "Average active sessions: the magic metric," Oracle USA, Tech. Rep.

[48] Hoopoes, "Oracle Database Tuning Statistics," acedido a: 25/03/2012. [Online]. Disponível em: http://www.hoopoes.com/cs/oracle_tune.shtml#recursive

[49] J. Maroco, *Análise estatística com utilização do SPSS*. Sílabo, 2010.

[50] M. Ghazanfari, M. Jafari, and S. Rouhani, "A tool to evaluate the business intelligence of enterprise systems," *Scientia Iranica*, no. 0, pp. –, 2011.

[51] H. J. Watson and B. H. Wixom, "The Current State of Business Intelligence," *Computer*, vol. 40, no. 9, pp. 96–99, 2007.

[52] S.-T. Li, L.-Y. Shue, and S.-F. Lee, "Business intelligence approach to supporting strategy-making of ISP service management," *Expert Syst. Appl.*, vol. 35, no. 3, pp. 739–754, 2008.

[53] M. Tereso and J. Bernardino, "Open source business intelligence tools for SMEs," in *Information Systems and Technologies (CISTI), 2011 6th Iberian Conference on*, 2011, pp. 1–4.

[54] C. Thomsen and T. Pedersen, "A Survey of Open Source Tools for Business Intelligence," in *Data Warehousing and Knowledge Discovery*, ser. Lecture Notes in Computer Science, A. Tjoa and J. Trujillo, Eds. Springer Berlin / Heidelberg, 2005, vol. 3589, pp. 74–84.

[55] PentahoCommunity, "Welcome to the Pentaho Community," acedido a: 15/12/2011. [Online]. Disponível em: http://community.pentaho.com

[56] Webdetails, "About | cde.webdetails.org," acedido a: 20/3/2012. [Online]. Disponível em: http://cde.webdetails.org/

[57] L. Mota, "Sistemas de Informação de Enfermagem: um estudo sobre a relevância da informação para os médicos," Mestrado de Informática Médica, Universidade do Porto, 2010.

[58] E. Ammenwerth, J. Brender, P. Nykänen, H.-u. Prokosch, M. Rigby, and J. Talmon, "Visions and strategies to improve evaluation of health information systems Reflections and lessons based on the HIS-EVAL workshop in Innsbruck," *International Journal of Medical Informatics*, 2004.

[59] S. Lameirão, "Gestão Hospitalar e o uso dos Sistemas de Informação: Aplicação ao CHVR-PR," Mestrado em Gestão, Universidade de Trás-os-Montes e Alto Douro, 2007.

[60] A. Abelha, J. Machado, V. Alves, and J. Neves, "Data warehousing through multi-agent systems in the medical arena," in *International Conference on Knowledge Engineering and Decision Support*, Porto, 2004.

[61] J. Machado, A. Abelha, P. Novais, J. Neves, and J. a. Neves, "Quality of service in healthcare units," *Int. J. Computer Aided Engineering and Technology*, vol. 2, no. 4, pp. 436–439, 2010.

[62] J. Machado, V. Alves, A. Abelha, and J. Neves, "Ambient intelligence via multiagent systems in the medical arena," *Engineering Intelligent Systems for Electrical Engineering and Communications*, vol. 15, no. 3, pp. 151–157, 2007.

[63] J. Machado, A. Abelha, J. Neves, and M. Santos, "Ambient intelligence in medicine," in *Biomedical Circuits and Systems Conference, 2006. BioCAS 2006. IEEE*, 29 2006-dec. 1 2006, pp. 94 –97.

[64] P. Silva, C. Quintas, M. Santos, A. Abelha, and J. Machado, "Step Towards Fault Forecasting in Hospital Information Systems," in *5th International Conference on BioMedical Engineering and Informatics*, Chongqing (China), 2012.

[65] P. Silva, C. Quintas, P. Gonçalves, G. Pontes, M. Santos, A. Abelha, and J. Machado, "Intelligent Systems based in Hospital Database Malfunction Scenarios," in *IEEE International Conference on Industrial Engineering and Engineering Management*, Hong Kong (China), 2012.

[66] P. Silva, J. Duarte, César Quintas, M. Santos, J. Neves, A. Abelha, and J. Machado, "Hospital Database Workload and Fault Forecasting," in *International Conference on Biomedical Engineering and Sciences*, Langkawi (Malásia), 2012.

Apêndice A

Relatório Nagios

Relatório Experimental sobre software Nagios

Paulo Silva
Informática Médica - Engenharia Biomédica
Universidade do Minho

Outubro de 2011

Conteúdo

Resumo

Este trabalho surge no âmbito da dissertação de Mestrado, denominada de "Monitorização e Prevenção de Falhas em Bases de Dados Hospitalares", de modo a aprofundar os conhecimentos sobre aplicações de monitorização.

O **Nagios** é um ferramenta de monitorização de componentes computacionais muito conhecida e com este relatório pretende-se ficar a conhecer um pouco melhor o seu funcionamento e em que medida pode ser utilizado para monitorizar bases de dados.

Apesar de ser relativamente fácil de utilizar é muito difícil proceder à sua instalação e configuração. Esta complexidade aumenta à medida que que se vão acrescentando componentes a monitorizar. Relativamente à monitorização de bases de dados, a versão gratuita do **Nagios** ainda não disponibiliza muitas funcionalidades para esse efeito.

Capítulo 1

Introdução

Para conhecer as soluções e funcionalidades disponibilizadas pela Nagios, acedeu-se ao site da empresa: **http://www.nagios.org/**. Neste site são apresentadas as duas principais soluções da **Nagios**. No separador **Enterprise** obtém-se informação sobre o **NagiosXI**, a solução comercial da **Nagios**. Por outro lado, no separador **Project**, obtém-se informação sobre a solução *open-source*, o **Nagios Core**.

1.1 NagiosXI

O **NagiosXI** é uma poderosa e popular ferramenta de monitorização de estruturas computacionais de uma organização.

Permite a monitorização dos vários equipamentos da rede da organização (hosts, switchs, servidores, impressoras) e possibilita a criação de alertas, caso algum parâmetro esteja fora dos valores normais.

Possui uma interface gráfica que facilita a sua utilização e leitura de resultados. Para além disto, a sua arquitetura permite que sejam adicionados outros módulos que disponibilizem outras funcionalidades.

Este software está disponível para download, em versões demo. Caso se pretenda obter na sua totalidade é necessário proceder à compra de uma licença de utilização. O preço da licença difere consoante o número de nodos da rede que se pretende monitorizar, aumentando à medida que o número de nodos a monitorizar aumenta.

1.2 Nagios Core

O software **Nagios Core** é um sistema *open-source* de monitorização de vários recursos, como por exemplo, de hosts, servidores, serviços e redes. Caso exista alguma anomalia nos parâmetros monitorizados, o software permite a criação de alertas para avisar o

administrador do sistema que algo está mal.

Este software possui uma arquitetura flexível e tem boa escalabilidade, o que possibilita a criação de outros projetos tendo por base o **Nagios Core**.

As principais funções deste software são então:

- Executar verificações - Verificar periodicamente o estado dos componentes.
- Manipulação de Eventos e Alertas - Análise dos eventos anormais e envio de avisos sobre os mesmos.
- Realização de controlos e notificações - É realizado o controlo da utilização dos recursos possibilitando a notificação de mensagens ao administrador.

A interface gráfica disponibilizada por este programa é uma interface web, que pode ser melhorada com a adição de novos *frontends*.

Este software está disponível para download na sua totalidade sem ser necessário pagar qualquer valor. No entanto, existe ainda a possibilidade de obter suporte durante um determinado tempo, sendo que é necessário pagar para usufruir desta funcionalidade.

1.3 Nagwin

Devido ao **Nagios Core** estar mais voltado para ser instalado em máquinas Unix, apesar de permitir monitorizar remotamente máquinas Windows, foi desenvolvido o projeto **Nagwin**, que não é mais do que uma versão do **Nagios Core** para Windows. Este software tem basicamente as mesmas funcionalidades que o **Nagios Core**, com a particularidade de ser extremamente fácil de instalar sendo apenas necessário correr o ficheiro executável.

Capítulo 2

Experimentação Nagios Core

O **Nagios Core** é apenas o núcleo do software **Nagios**. Por conseguinte, para que se possa colocar este software em funcionamento foi necessário instalar mais alguns componentes, nomeadamente, *plugins*, servidor *apache* e outras bibliotecas que ajudam na instalação.

2.1 Instalação

Numa primeira fase, foi realizada a instalação manual do **Nagios Core** seguindo o tutorial disponível em: http://www.vivaolinux.com.br/artigo/Instalando-o-Nagios-Core-3.2
+Nagios-Plugins+NRPE+NSClient+++pnp4nagios+FrontEnd-no-Ubuntu-10.4-LTS

No entanto, devido a alguns problemas que surgiram na configuração, houve necessidade de procurar uma nova forma de instalar. No site da **Nagios**, encontram-se também vários *addons* (projetos relacionados com o **Nagios** desenvolvidos por outras pessoas). Um desses *addons*, é um *script* de instalação do **Nagios Core**, disponível em:
http://exchange.nagios.org/directory/Addons/Installation/Nagios-Auto-Installation-Script/details

Este script simplifica o processo de instalação e configuração, pois realiza automaticamente o download do **Nagios Core** e respetivos *plugins*, procedendo no final à instalação dos mesmos.

Durante o processo de instalação foi criado um utilizador "nagiosadmin" com a password "paulo89". Este login será necessário para fazer a autenticação na interface web do programa.

2.2 Configuração

Após a instalação é necessário configurar alguns ficheiros:

- **commands.cfg** - é responsável por ligar os comandos enviados pelos serviços de verificação, com os *plugins* presentes no servidor **Nagios**, por exemplo:

```
Define command{
   command_name  check_local_load
   command_line  $USER1$/check_load -w $ARG1$ -c $ARG2$
              }
```

- **timeperiods.cfg** - Aqui são definidos os períodos de tempo nos quais devem ser executadas as verificações, por exemplo:

```
# TEMPO INTEGRAL 24x7
define timeperiod{
        timeperiod_name       24x7
        alias         24 Hours A Day, 7 Days A Week
        sunday              00:00-24:00
        monday              00:00-24:00
        tuesday             00:00-24:00
        wednesday           00:00-24:00
        thursday            00:00-24:00
        friday              00:00-24:00
        saturday            00:00-24:00
               }
```

- **contacts.cfg** - Neste ficheiro é definido o destinatário das notificações, por exemplo:

```
define contact{contact_name  nagios
      use                   generic-contact
      alias                 Nagios Admin
      email                 silva.paulojorge@hotmail.com
}
```

- **localhost.cfg** - São definidos, entre outras coisas, os serviços de verificação que devem correr nesta máquina, por exemplo:

```
define service{use             local-service
      host_name                localhost
      service_description      PING
  check_command check_ping!100.0!20!60%
}
```

Os ficheiros mencionados acima são os essenciais para o **Nagios** monitorizar o *localhost*. No entanto, caso seja para aplicar a vários componentes, outros ficheiros teriam de ser configurados, como por exemplo:

- **printer.cfg** - para configurar as impressoras a monitorizar.
- **switch.cfg** - para configurar os switchs a monitorizar.

Após proceder à configuração e ao arranque do servidor Nagios, a aplicação gráfica fica disponível em **http://localhost/nagios**, como se pode observar na Figura 1 do Apêndice A.

2.3 Serviços de verificação instalados

Foram utilizados os ficheiros exemplo de configuração sugeridos pela **Nagios**, pelo que os serviços disponibilizados por estes são (consultar Figura 2 do Apêndice A):

- **Current Load** - Verifica a carga do processador
- **Current Users** - Verifica quantos utilizadores estão activos na máquina num determinado momento.
- **HTTP** - Verifica se o serviço http (Hypertext Transfer Protocol) está a funcionar.
- **PING** - Verifica se consegue comunicar com outros hosts, efetuando para tal um ping.
- **Root Partition** - Verifica o disco, dando a informação do espaço disponível.
- **SSH** - Verifica a disponibilidade do serviço ssh (secure shell).
- **Swap Usage** - Verifica a memória swap que é utilizada.
- **Total Processes** - Verifica e dá a informação de quantos processos estão a correr no momento da verificação.

Para além dos serviços de verificação, o **Nagios Core** possui ainda as seguintes funcionalidades:

- Agendamento de serviços de verificação.
- Notificações, via e-mail, caso algum dos componentes que se está a monitorar esteja com um problema.
- Permite obter relatórios sobre a disponibilidade do sistema que se está a monitorizar.

Capítulo 3

Adição de novas funcionalidades

Este software tem a vantagem de possibilitar a adição de novas funcionalidades através da instalação de *plugins*. No site da **Nagios** encontram-se bastantes *plugins*, no entanto, a instalação deles revela-se um pouco complexa e são poucos os que têm uma página com suporte adequado.

3.1 Adição de uma interface gráfica mais amigável

A aparência por defeito do **Nagios Core** não é muito agradável. Para melhorar este facto a **Nagios** disponibiliza no seu site alguns *frontends* que podem ser instalados em cima do **Nagios Core** melhorando a sua interface web. O *frontend* escolhido para teste foi o **Nuvola**. A instalação deste componente revelou-se mais uma vez uma tarefa complexa, apesar dos passos estarem mencionados num ficheiro "readme" no pacote de instalação.

3.2 Melhorar a visualização de resultados

Os resultados provenientes dos serviços de verificação do **Nagios** estão no formato numérico, dificultando a visualização do desenrolar de determinado parâmetro ao longo do tempo. Uma forma de melhorar esta visualização é transpor os dados numéricos para gráficos. Para tal foi necessário instalar o programa **pnp4nagios**, que é uma interface responsável por, a partir dos dados recolhidos pelo **Nagios**, criar gráficos de desempenho, permitindo ao utilizador uma melhor perceção do que vai acontecendo ao longo do tempo. Na Figura 3 do Apêndice A encontra-se um exemplo da utilização desta aplicação.

Para a instalação deste pacote foi necessário seguir o tutorial disponível em:
http://nagiosnapratica.wordpress.com/2010/12/01/artigo-11-grafico-no
-nagios-com-pnp4nagios/

Capítulo 4

Monitorização de base de dados com o Nagios

Monitorizar o estado de uma base de dados é uma tarefa muito complexa, devido a este facto, torna-se importante a utilização de ferramentas como o **Nagios**, de forma a automatizar e simplificar este processo.

4.1 Armazenamento dos dados capturados pelo Nagios

Para que se possam executar ações de análise é necessário proceder ao armazenamento dos dados recolhidos pelo **Nagios** aquando a execução dos vários testes. Essa informação pode ser armazenada em base de dados. No site da *Nagios* encontram-se vários *addons* que possibilitam a integração do **Nagios** com distintas bases de dados. São exemplos: o **NDOUtils** e **Nagdb** que permitem guardar/exportar os dados recolhidos para uma base de dados MySQL.

4.2 Plugins de monitorização

O **Nagios** faz uso de um conjunto de *plugins* para realizar as tarefas de monitorização salientando-se os seguintes:

- **Check_tcp** - Teste básico responsável por apurar se a porta de ligação da base de dados à internet está aberta por forma a possibilitar a comunicação das diferentes aplicações com a base de dados. Em bases de dados Oracle a porta pré-definida costuma ser a TCP 1521 e em base de dados MS SQL Server a TCP 1433.

- **Check_oracle** - Verifica se determina instância da base de dados está a correr e se é possível fazer login. Para executar este *plugin* é necessário ter um cliente Oracle

instalado.

- **Check_dbversion_oracle.java** - Fornece informações sobre as bases de dados presentes no sistema.
- **check_tablespace_oracle.java** - Fornece informações relativas ao espaço existente nas *tablespaces*.
- **Check_oracle_health-1.6.7.tar.gz** - Fornece informação sobre alguns parâmetros importantes como a utilização da cache, uso de indexes e e informações relativas à tabela de *locks*.

NOTA: Devido à existência de distintos vendedores e vários tipos de base de dados, a maioria dos *plugins* é escrita em Java permitindo assim uma implementação mais fácil. No entanto, para correr estes *plugins* é necessário instalar os *drivers* da base de dados correspondente, no caso da Oracle, o *Oracle-jodbc*.

Existem versões destes *plugins* para base de dados de diferentes empresas, basta para tal consultar o artigo presente no site:

http://nagios.frank4dd.com/howto/db-monitoring.htm

Capítulo 5

Conclusões

Após pesquisa e realização de alguns testes, constatou-se que o **Nagios Core** é um software de fácil utilização que apresenta ferramentas bastante úteis para monitorizar equipamentos, às quais podem ser adicionados ainda *plugins* para implementar novas funcionalidades. Este software consegue monitorizar diversos equipamentos (hosts, impressoras, servidores, switchs, base de dados), permitindo agendar verificações, ou seja, permite realizar as verificações num determinado momento do dia que não esteja ninguém na organização. Para além disto, possibilita o envio de notificações via email quando alguma anomalia é detetada.

No entanto, verificou-se que processo de instalação/configuração é complexo, apresentando pequenas nuances dependendo do sistema operativo onde se pretende instalar. Esta mesma dificuldade verifica-se na instalação de alguns *plugins* devido à falta de suporte adequado. Outra tarefa de grande complexidade é a adição de novos servidores/serviços, pois exige uma configuração mais completa, sendo necessário modificar os ficheiros base de configuração. Para tal, é necessário possuir alguns conhecimentos de comandos Unix. Em relação à monitorização de bases de dados, esta é limitada pelo pequeno número de *plugins* existentes para o efeito.

Em suma, o sistema **Nagios Core** acompanhado com outros programas (*plugins, pnp4nagios*) revela-se útil e eficaz nas tarefas de monitorização que se encontram configuradas. Todavia, a complexidade de instalação de novas funcionalidades e de configuração de outros equipamentos pode levar a um elevado custo a nível de tempo.

Apêndice A

Imagens do Nagios

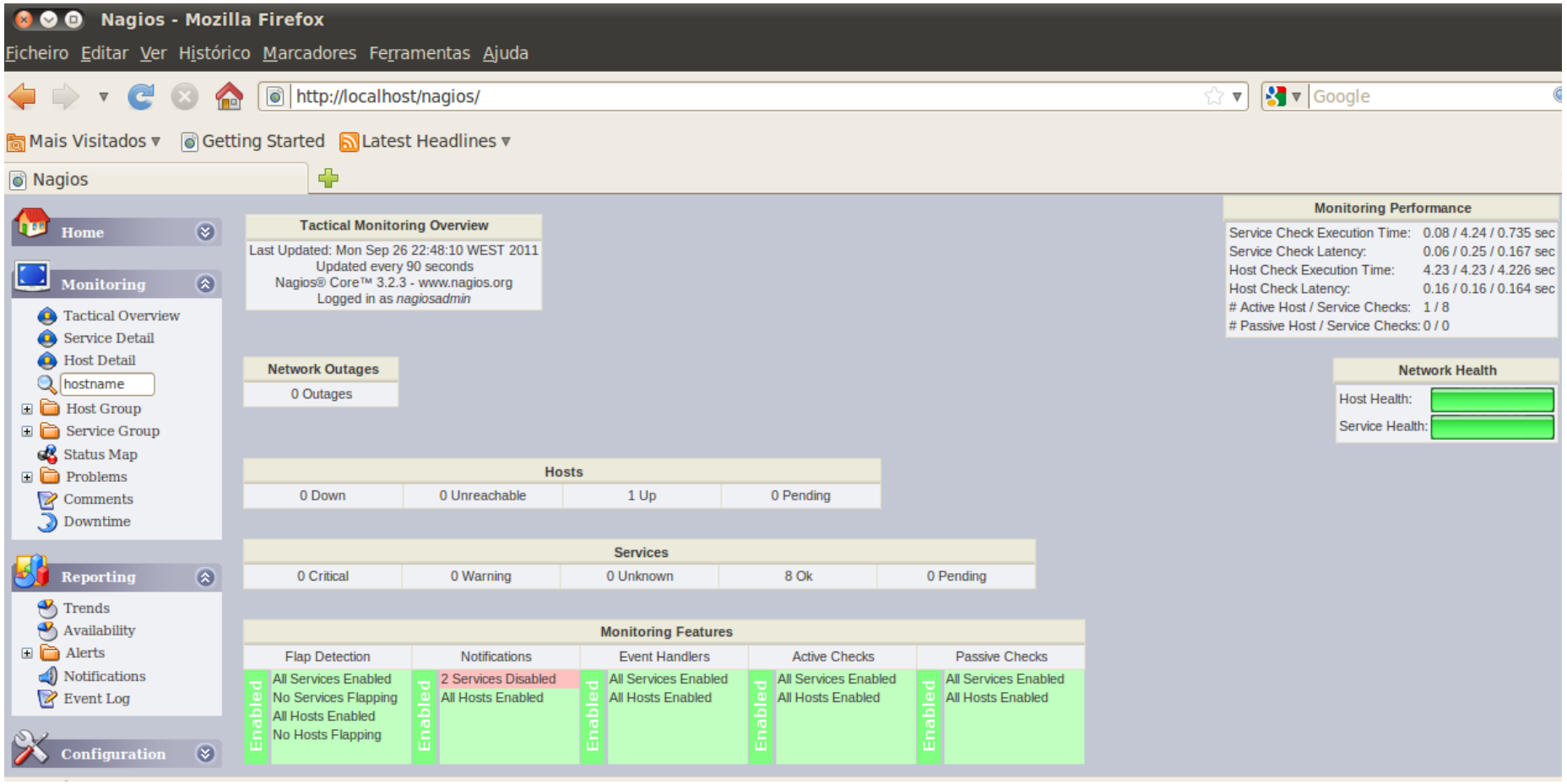


Figura 1: Página inicial do **Nagios** utilizand o *frontend* **Nuvola**

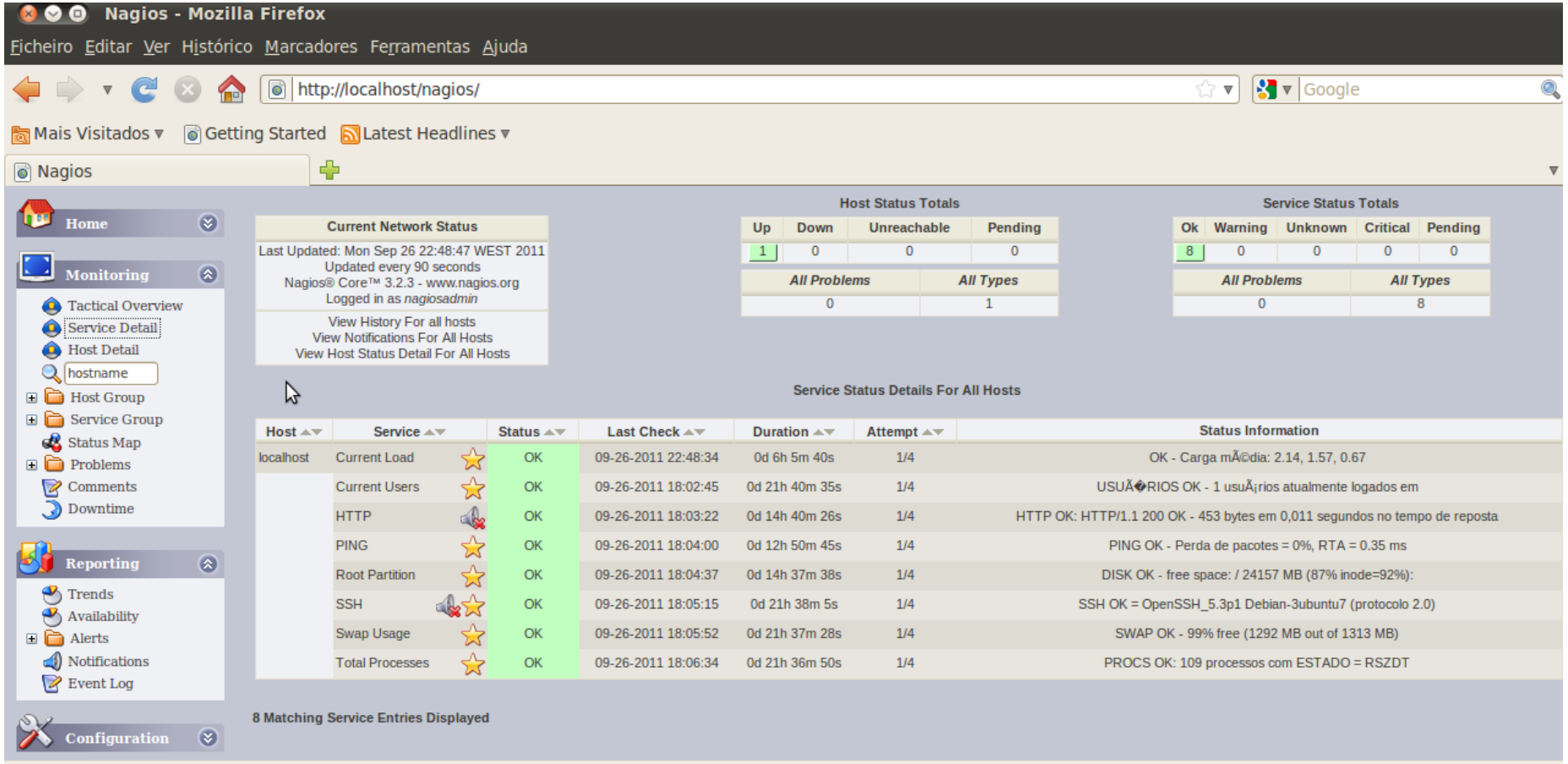


Figura 2: Página inicial dos serviços de verificação a correr no *localhost*

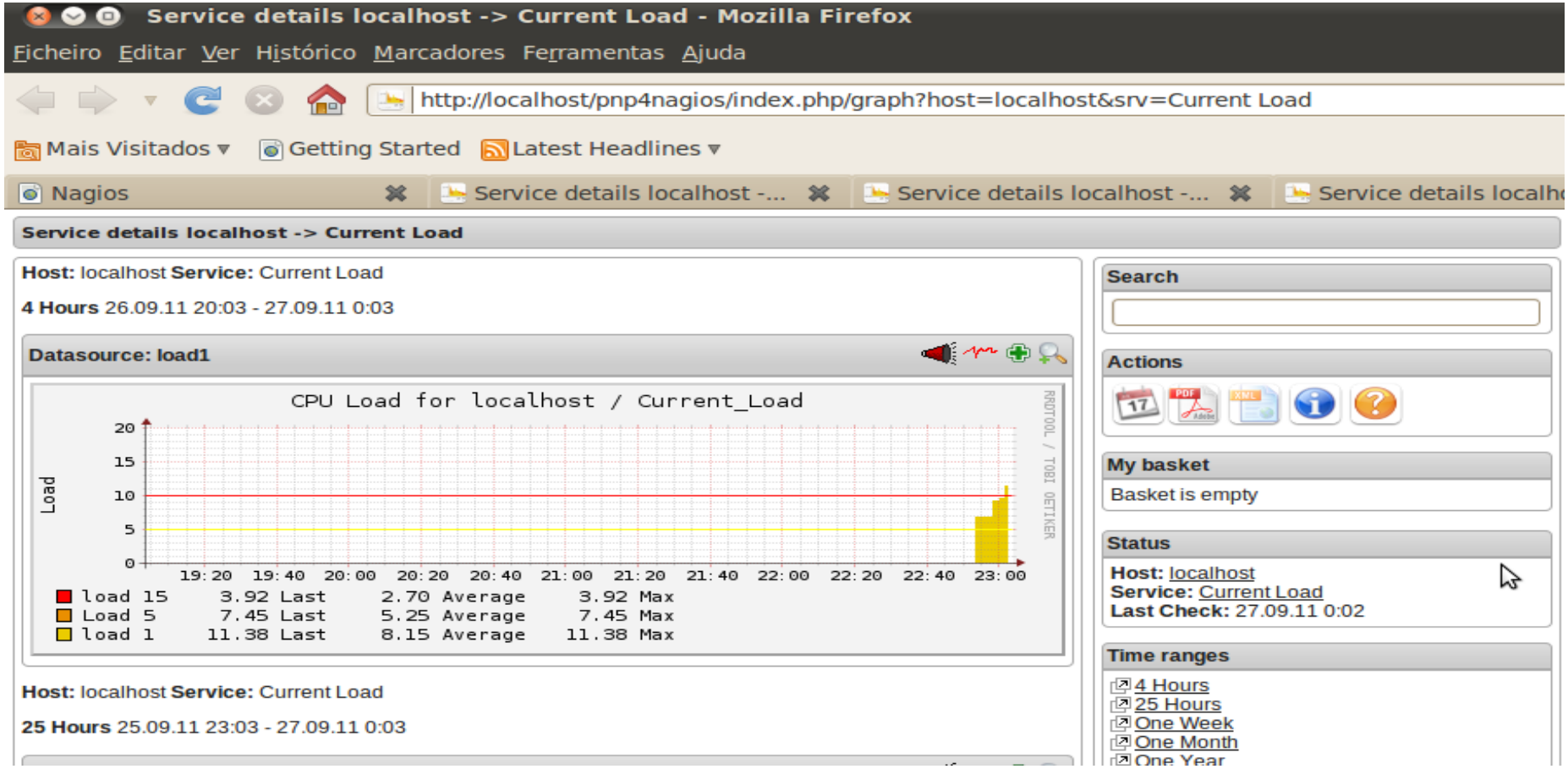


Figura 3 : Exemplo de aplicação do **pnp4nagios**

Apêndice B

Manual Pentaho

Manual de Instalação/Utilização do Pentaho Community

Paulo Silva
Informática Médica - Engenharia Biomédica
Universidade do Minho

Março de 2012

Conteúdo

Lista de Figuras

Resumo

Este trabalho surge no âmbito da dissertação de mestrado denominada de "Monitorização e Prevenção de Falhas em Bases de Dados Hospitalares", mais concretamente, na parte da exploração de ferramentas de Business Intelligence.

Após a realização de alguma investigação e algumas experiências apurou-se que a ferramenta Pentaho Community é eficaz no tratamento de grandes quantidades de dados, promovendo uma clara visualização e interpretação dos mesmos.

Este manual tem como objetivos indicar os principais passos para a instalação desta versão do Pentaho e exemplificar algumas das suas funcionalidades. Serão abordados com maior pormenor os módulos responsáveis pela criação de dashboards e integração de dados.

Capítulo 1

Introdução

Nas grandes organizações a quantidade de dados armazenada é bastante elevada, verificando-se em muitas delas a utilização de ferramentas que permitem o tratamento (obtenção, análise e manutenção) dos dados e a extração de informações relevantes para a organização. A este conjunto de ferramentas dá-se o nome de ferramentas de Bussiness Intelligence [1, 2].

A utilização deste tipo de ferramentas nesta dissertação é justificada, pelo facto da quantidade dos dados relativos ao processo de monitorização das bases de dados ser bastante elevada, tornando difícil a sua interpretação. Após a leitura de alguns estudos sobre ferramentas *open-source* de Bussiness Intelligence, constatou-se que a versão Community, do Pentaho, era a mais indicada para os objetivos propostos [2, 3].

A ferramenta Pentaho foi lançada no ano de 2004 e possui uma versão paga (Enterprise) e uma gratuita denominada por Community. A versão gratuita permite a criação de relatórios, gráficos e dashboards; a aplicação de técnicas de data mining e a exportação de dados. Esta ferramenta é constituída por várias aplicações todas elas escritas em Java, o que aumenta a sua portabilidade. Contém um módulo principal, o *bi-plataform*, ao qual podem ser adicionados módulos extra responsáveis por disponibilizar outras funcionalidades [2, 4]. De entre os módulos disponíveis salientam-se:

- **Community dashboard Editor (CDE)** - Permite o desenvolvimento de dashboards, para mostrar de uma forma mais amigável os dados recolhidos/tratados. Foi desenvolvido pela Webdetails. O layout do dashboard é baseado em HTML (HyperText Markup Language) e a ele podem ser adicionados vários componentes, tais como: tabelas, gráficos, imagens e caixas de texto [4, 5]. Os dados que são mostrados em cada componente provêm de vários tipos de fontes de dados, desde queryies SQL (Structured Query Language) até ficheiros de XML(eXtensible Markup Language) [4].
- **Pentaho Report Designer** - Permite a criação de relatórios de uma forma fácil

e intuitiva. Esta é uma ferramenta muito completa permitindo até a inclusão de gráficos nos relatórios [4].

- **Pentaho Data Integration (Spoon)** - Permite a extração, integração e transformação de dados provenientes de diferentes fontes [4].
- **Mondrian** - Permite o processamento da informação histórica armazenada através de técnicas de OLAP (Online Analytical Processing) [4].
- **Weka** - Permite a aplicação de técnicas de data mining sobre os dados. Note-se que o Weka não é um módulo do Pentaho, mas sim uma outra ferramenta também ela gratuita, que pode ser interligada com o Pentaho através do plugin "Weka Scoring Plugin" no módulo Spoon [4].

Numa fase inicial, o módulo CDE foi o mais explorado, uma vez que o objetivo era a criação de dashboards que permitissem visualizar de uma forma mais clara os dados recolhidos. Com o CDE a interpretação dos dados e a extração de possíveis padrões tornaram-se tarefas bem mais fáceis. Posteriormente foi utilizado o Spoon para simplificar as operações de manipulação de dados mais complexas.

Este manual encontra-se dividido em três partes. Na primeira encontram-se os principais passos da instalação da plataforma Pentaho Community e dos módulos CDE e Spoon. Na segunda parte, encontram-se informações relativas à criação de dashboards. Por último, na terceira parte são apresentados alguns exemplos de transformações utilizando o Spoon.

Capítulo 2

Instalação

O processo de instalação do módulo *bi-plataform* é relativamente fácil, sendo necessário apenas como pré-requisito possuir o Java na máquina onde se deseja instalar. De seguida, é necessário fazer o download da versão mais recente do Pentaho Community. No momento da realização deste manual, a versão mais recente é a *biserver-ce-3.9.0-stable*, estando disponível em vários formatos para as várias plataformas computacionais no link:

http://sourceforge.net/projects/pentaho/files/Business%20Intelligence%20Server/3.9.0-stable/

Após efetuar o download deve-se descomprimir os ficheiros para uma pasta que será o diretório raiz do Pentaho, esta pasta pode ser denominada por Pentaho. O próximo passo consiste em definir as variáveis de ambiente às quais o Pentaho tem de aceder para funcionar, essas variáveis são:

- **CATALINA HOME** - deve ser indicado o caminho para o diretório do servidor apache-tomcat (situa-se dentro da pasta do Pentaho).
- **JAVA HOME** - deve ser indicado o caminho para o diretório da versão *jdk* do Java.

Em máquinas Windows para aceder às variáveis ambiente devem-se seguir os passos:

1. Premir o botão iniciar, "Botão direito" em cima de "O meu computador", Selecionar "Propriedades";
2. No painel que surge deve-se clicar em "definições avançadas do sistema";
3. No novo painel clicar no separador "Avançadas" e de seguida no botão "Variáveis de ambiente";
4. Utilizar os botões disponíveis para adicionar estas duas variáveis e respetivos caminhos.

Após estas configurações o servidor Pentaho está pronto para ser utilizado. Pode ser executado de duas formas:

- Através de duplo clique no ficheiro *start-pentaho.bat*.
- Através da linha de comandos: *C:\pentaho\biserver-ce\start-pentaho.bat*

Para além do servidor, convém também executar a aplicação administrativa (Pentaho Administration Control-PAC). Esta aplicação permite fazer a gestão de recursos, como por exemplo, adicionar novos utilizadores e respetivos privilégios ou adicionar novas fontes de dados para análise. Para iniciar o modo administrativo:

- Duplo clique no ficheiro *start-pac*.
- Através de *C:\pentaho\administration-console\start-pac.bat*.

Depois de inicializar estes dois componentes, o acesso às aplicações é efetuado através do browser. Se pretender aceder à aplicação Pentaho deve-se colocar o seguinte endereço:

http://localhost:8080/pentaho

Após a realização deste passo aparecerá no browser o painel de login do Pentaho onde é necessário efetuar a autenticação para começar a utilizar o programa, este painel pode ser visualizado na Figura 2.1.

Figura 2.1: Página de login da ferramenta Pentaho

No caso de desejar aceder às configurações administrativas deve colocar este endereço:

http://localhost:8099/

Aqui também é necessário efetuar o login. As credenciais definidas por defeito são: **username:** *admin* e **password:** *password*. Visto que a aplicação administrativa é muito importante, pois contém informações sobre utilizadores e respetivas políticas de privacidade, deve-se proceder à mudança das credenciais o mais rapidamente possível. Para tal, é necessário proceder às seguintes operações [6]:

1. Proceder à encriptação da nova password através do comando: *java -cp lib/jetty-6.1.2.jar;lib/jetty-util-6.1.9.jar org.mortbay.jetty.security.Password admin mypassword*
2. Copiar a password encriptada para o ficheiro login.properties. que se localiza em: C:\pentaho\administration-console\resource\config, substituindo a antiga.
   Password encriptada OBF:1v2j1uum1xtv1zej1zer1xtn1uvk1v1v
3. Reiniciar a aplicação administrativa e efetuar o login com a nova password.

Estes comandos são válidos para sistemas operativos Windows. Para outros sistemas deve-se procurar documentação sobre comandos equivalentes.

É ainda necessário adaptar o Pentaho à base de dados com a qual irá trabalhar. Se a base de dado em questão for Oracle deve-se copiar o driver *ojdbc14.jar* para a pasta: C:\\pentaho\administration-console\jdbc\.

2.1 Instalação do CDE

O CDE (Community Dashboard Editor) é um editor de dashboards do Pentaho, que é instalado sobre o *bi-plataform* do Pentaho Community [4]. Devido à grande utilização deste módulo tem sido feito um esforço contínuo na tentativa de o aperfeiçoar. No início deste estudo, a sua instalação tinha de ser feita passo-a-passo. Atualmente, encontra-se já desenvolvido um script que facilita a instalação deste módulo.

2.1.1 Instalação Manual

O primeiro passo para a instalação deste módulo é fazer o download do pacote que contém os ficheiros de instalação, através do link:

http://code.google.com/p/cdf-de/

Depois de se obter o pacote procede-se à descompressão do mesmo, obtendo-se assim um conjunto de pastas. Após isto, deverá copiar as pastas *bi-developers, cde_sample, cdf, CST* para o diretório onde o Pentaho foi previamente instalado, mais precisamente em:

C:\pentaho\biserver-ce\pentaho-solutions

Por último, deve-se copiar as pastas *cda, pentaho-cdf, pentaho-cdf-dd,* que se encontram na pasta *system* do pacote de instalação do CDE, para o diretório:

C:\pentaho\biserver-ce\pentaho-solutions\system

Para além disto, é ainda necessário acrescentar o driver *ojdbc14.jar* à pasta na qual se irão desenvolver os dashboards, caso estes necessitem de informação que esteja armazenada em bases de dados Oracle.

Após a reinicialização do servidor Pentaho, o módulo CDE já estará disponível quer através do menu, quer através de um botão na parte superior da janela do Pentaho.

2.1.2 Instalação Automática

A Webdetails, empresa responsável pelo desenvolvimento e manutenção da ferramenta CDE, disponibiliza um instalador automático. Para sistemas Unix é trivial, pois apenas é necessário correr o script disponibilizado em:

http://pedroalves-bi.blogspot.pt/2011/12/back-to-basics-step-by-step-pentaho.html

Para ambientes Windows, é um pouco mais trabalhoso, uma vez que é preciso instalar o cygwin. O cygwin é um programa utilizado para simular uma bash em máquinas Windows. Após a instalação deste programa, é possível executar o script, para tal deve-se seguir o tutorial presente em:

http://codeissue.com/articles/a04e87158bb8552/pentaho-bi-ctools-cdf-cda-cde-saiku-analytics-etc-using-cygwin

Este tutorial pode ser resumido em dois passos, nomeadamente:

- Instalar o cygwin acionando, durante a instalação, a incorporação dos comandos: wget, unzip.
- Copiar o script e executá-lo através:
  .\ctools-installer.sh -s \cygdrive\c\Users\Paulo\pentaho\biserver-ce-3.9.0-RC1\ biserver-ce\pentaho-solutions\

Este script tem a vantagem de instalar a versão mais atual do CDE, que apresenta, entre outras funcionalidades, uma maior gama de gráficos que podem ser utilizados.

2.2 Instalação do Pentaho Data Integration

Esta aplicação revela-se muito importante para a integração dos dados. É muitas vezes utilizada para realizar operações entre dados provenientes de diferentes fontes. Para além disso, pode ser usada para dividir operações complicadas em pequenos passos mais fáceis de realizar. Para proceder à instalação deste módulo apenas é necessário obter o ficheiro executável, que pode ser encontrado no link:

http://sourceforge.net/projects/pentaho/files/Data%20Integration/4.2.1-stable/

O ficheiro resultante deve ser descomprimido para uma pasta, de preferência junto ao local onde se tem o módulo principal instalado. Por fim, para iniciar a aplicação deve-se fazer duplo clique sobre o ficheiro spoon.bat.

Capítulo 3

Utilização do Pentaho

Neste capítulo serão apresentadas e explicadas algumas funcionalidades que a ferramenta Pentaho oferece, nomeadamente nos módulos CDE e Spoon que foram os mais utilizados na execução da dissertação. Inicialmente, será apresentado o menu inicial do Pentaho, sendo logo após, explicado processo de criação de dashboards. Por fim, será apresentado o processo de integração de dados usando a ferramenta Spoon.

3.1 Menu Inicial Pentaho

Para iniciar o Pentaho deve-se efetuar o procedimento mencionado acima. Depois de efetuar o login abre-se a página inicial do Pentaho, como pode ser observado na Figura 3.1.

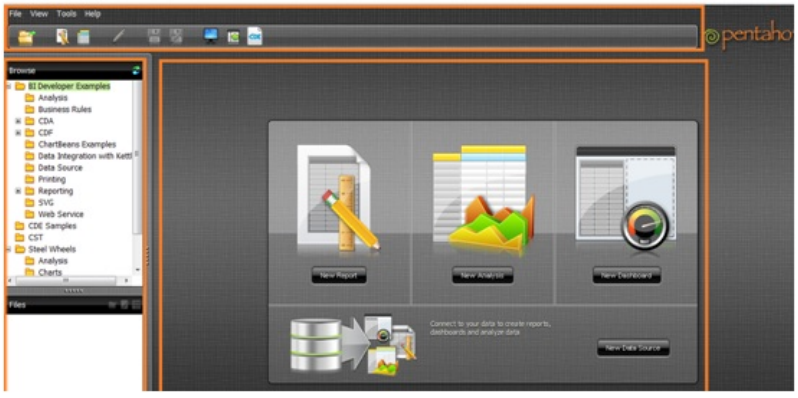


Figura 3.1: Página inicial do Pentaho

Esta página está dividida em três áreas principais. A barra de menus, que pode ser encontrada na parte superior da página, onde se encontram os ícones que permitem aceder às diversas funcionalidades Pentaho. A parte central, que será a zona reservada para a área de trabalho, após a escolha do tipo de projeto que se pretende desenvolver. Na zona lateral

da página inicial, encontra-se um pequeno browser que permite ao utilizador navegar pela hierarquia de pastas associadas ao Pentaho. Como se pode observar na Figura 3.1, existe a possibilidade de a partir do servidor principal criar relatórios e efetuar análises OLAP. No entanto, as ferramentas disponibilizadas por este são algo limitadas. Para quem necessitar de usar estas funcionalidades torna-se vantajoso instalar os módulos adicionais: Pentaho Report Designer e Mondrian.

3.2 Utilização do CDE

Para aceder ao separador de elaboração de dashboard deve-se carregar no separador CDE, como está indicado na Figura 3.2.

Figura 3.2: Ícone do CDE

Abre-se então a àrea de trabalho relativa à elaboração de dashboards, que pode ser visualizada na Figura 3.3. Na parte superior da área de trabalho encontram-se os seguintes itens:

- New - Serve para criar um novo dashboard.
- Save - Quando é necessário guardar as alterações efetuadas no projeto de elaboração do dashboard.
- Save as...- Se pretender guardar o projeto com outro nome ou em outro local deve-se utilizar este separador.
- Reload - Para fazer o reset do dashboard, todas as operações efetuadas até ao momento são anuladas.
- Settings - No caso de pretender alterar algumas configurações, como por exemplo, definição do autor.
- Layout - Neste separador encontram-se as ferramentas para desenvolvimento do layout da página de dashboard.
- Components - Neste separador encontram-se os vários componentes que se podem adicionar a um dashboard, nomeadamente: tabelas, gráficos, caixas de seleção, parâmetros, etc.
- Data Source - Separador onde é necessário definir as fontes de onde são extraídos os dados para a construção dos dashboards.
- Preview - Separador que promove a pré-visualização das alterações efetuadas até ao momento.

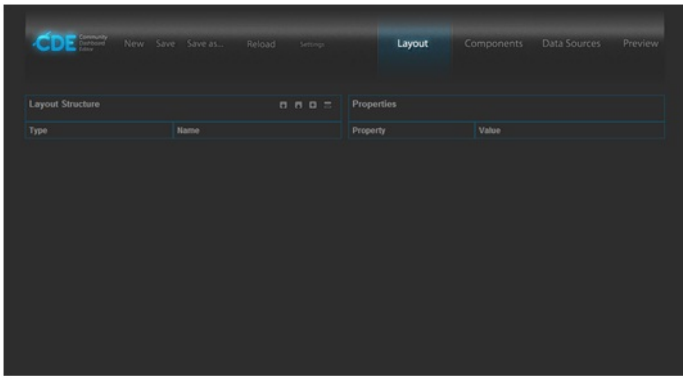


Figura 3.3: Área de trabalho do módulo CDE

3.2.1 Criação do Layout

Um dos primeiros passos na criação de um dashboard é a definição do layout (estrutura). Este processo consiste em montar uma estrutura de página adicionando para esse efeito um conjunto de recursos, tais como, linhas, colunas, espaços, imagens e elementos HTML. Os botões associados a cada um destes elementos, encontram-se descritos na Figura 3.4.

Figura 3.4: Botões de edição do layout

Cada elemento tem bastantes parâmetros que podem ser alterados, como por exemplo, altura, largura, cor, etc. No entanto, existe um elemento especial: o "adicionar recursos", este elemento é utilizado para criar um ficheiro com o estilo de formatações pretendido para as *tags* HTML. Por exemplo, definir o tamanho de letra da *tag* "<h1>", após estar especificado neste ficheiro, sempre que se utilizar "<h1>", este fica com o tamanho de letra definido no ficheiro, evitando assim múltiplas formatações.

A definição da estrutura é importante pois permite localizar espacialmente os objetos que compõem o dashboard, por exemplo, após a criação de um gráfico é possível definir uma determinada linha e/ou coluna onde este se deve situar. Após realizar alterações é

possível carregar no botão de pré-visualização e verificar as modificações elaboradas no dashboard.

3.2.2 Adição de fontes de dados

Um dashboard é uma forma simples e atrativa de mostrar um conjunto de dados, por isso, o elemento fundamental dos dashboard são as fontes de dados. O CDE permite adicionar vários tipos de fonte de dados, nomeadamente a partir de vários tipos de queries: SQL, MDX, Scripting, OLAP4J, XPATH, KETTLE, MQL. No entanto, neste tutorial serão abordadas as queries SQL, uma vez que foi a interface escolhida para aceder à base de dados Oracle e as queries Kettle que estão relacionadas com o módulo Spoon.

Na barra lateral, seleciona-se queries, seguido de *sql over sqlJdbc* (tipo de driver utilizado para estabelecer a conexão com a base de dados) e preenche-se o formulário apresentado na Figura 3.5.

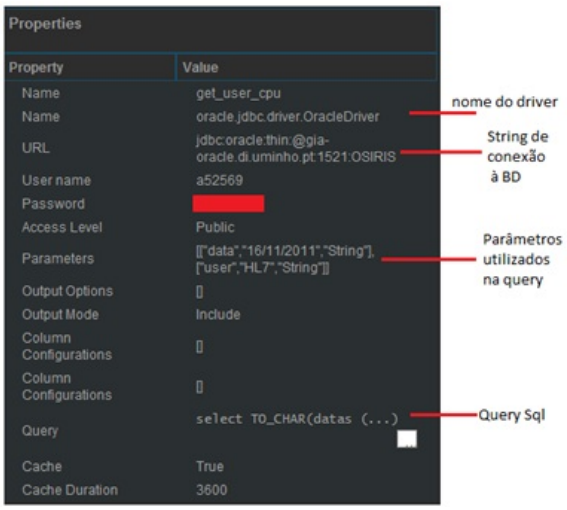


Figura 3.5: Propriedades fonte: query SQL

No caso de se pretender usar uma query Kettle, na barra lateral seleciona-se *Kettle query* e de seguida em *kettle over kettleTransFromFile*. Deve-se preencher os atributos de acordo com a Figura 3.6.

3.2.3 Adição de componentes

São vários os componentes que se podem inserir num dashboard, no entanto, os mais comuns são tabelas, gráficos e caixas de seleção, por isso apenas serão exemplificados os passos para a criação destes componentes. Todavia, a interface amigável do CDE torna fácil a inserção dos outros componentes.

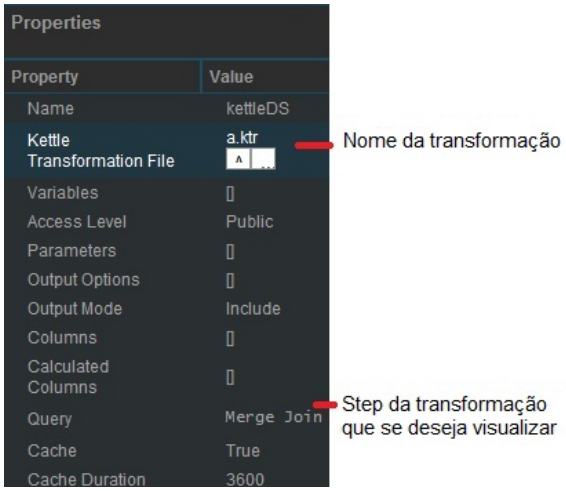


Figura 3.6: Propriedades fonte: query Kettle

Para adicionar caixas de seleção, basta clicar na barra lateral na opção "Selects" e escolher a opção pretendida, neste caso "select component". As caixas de seleção permitem atualizar outros componentes, de acordo com o parâmetro selecionado. É necessário preencher as seguintes propriedades da Figura 3.7 e criar um novo componente "Simple Parameter" para que o parâmetro da caixa de seleção seja global a todo dashboard.

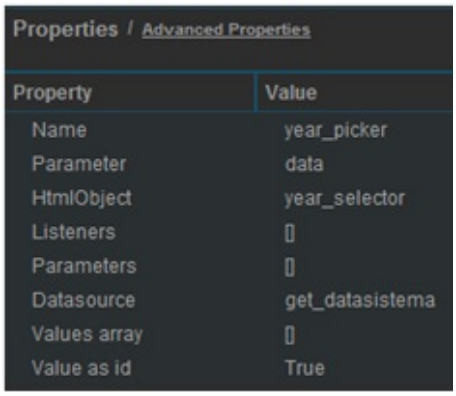


Figura 3.7: Propriedades do select component

Para adicionar um gráfico, seleciona-se a opção "Charts", na barra lateral, e procede-se à escolha do tipo de gráfico: barras, linhas, pizza, tempo. Seguidamente, deve-se preencher o painel relativo às propriedades do gráfico.

Este painel está dividido em duas partes, a primeira corresponde às propriedades básicas do gráfico como definição de altura, largura etc. A segunda parte é onde se definem as propriedades avançadas, tais como, a especificação de "extension points". Com esta propriedade é possível alterar através de código, algumas características dos gráficos que o programa não permite efetuar graficamente. Na tabela 3.1, encontram-se alguns exemplos.

Tabela 3.1: Extension Points

Nome	Valor	Descriçãoo
xAxisLabel_textAngle	45	Rotação de 45 graus da label do eixo dos x
yAxisLabel_textAngle	45	Rotação de 45 graus da label do eixo dos y
xAxisLabel_font	12px Arial	Muda o tamanho e o tipo de letra do eixo x
yAxisLabel_font	12px Arial	Muda o tamanho e o tipo do eixo y
yAxisLabel_text	function(d)return d + 'MB'	Acrescenta "MB" aos valores de y
xAxisLabel_text	function(d)return d.toFixed(2)	Fixa duas casas decimais aos valores de x
titleLabel_text	function() return user	Muda o titulo do gráfico com base no parâmetro user

Na Figura 3.8, encontram-se definidas algumas propriedades para um gráfico de linhas que tem por base a fonte de dados criada anteriormente. Pode-se aceder às propriedades avançadas carregando no separador destinado às mesmas.

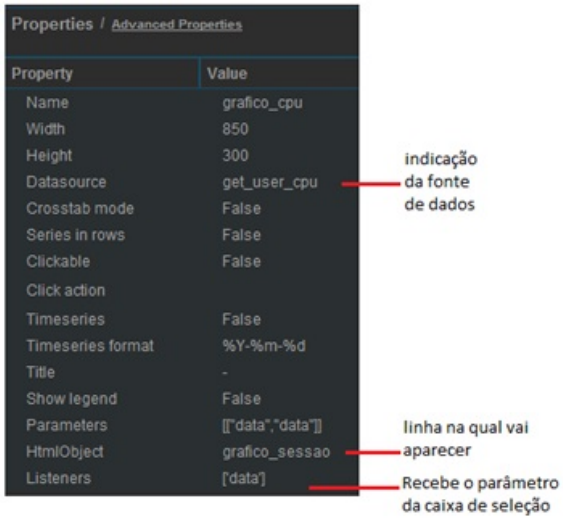


Figura 3.8: Propriedades do gráfico

Para adicionar tabelas, o procedimento é similar aos componentes anteriores. Primeiro carregar na barra lateral em "others" e depois selecionar "table component". Mais uma vez, é preciso definir um conjunto de propriedades relacionadas com o conteúdo e formatação da tabela. Na Figura 3.9, encontra-se um exemplo das propriedades de uma tabela.

As propriedades avançadas da tabela podem ser acedidas da mesma forma que fora utilizada para aceder às do gráfico.

Uma opção muito importante quando os dashboards pretendem mostrar a informação em tempo real é a questão da atualização. Para atualizar um componente é necessário acrescentar em *pre execution* a função que vai efetuar o *refresh* desse componente: *func-*

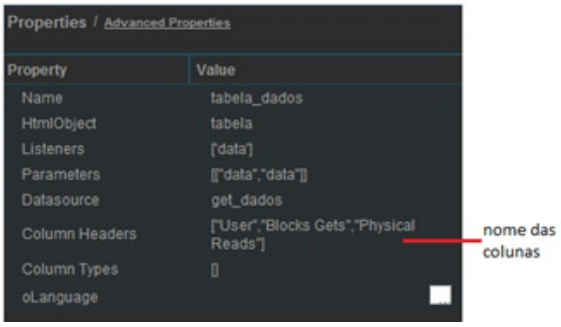


Figura 3.9: Propriedades da tabela

tion() this.refreshPeriod = 5 . Neste caso, o componente seria atualizado de cinco em cinco segundos.

3.3 Utilização do Pentaho Data Integration

Para iniciar esta ferramenta deve-se aceder à pasta onde foi instalada e clicar no ficheiro *spoon.bat*. Da primeira vez que este programa é executado, aparece um formulário onde é necessário definir o nome de utilizador e o nome do repositório desse utilizador. O repositório é a pasta onde posteriormente serão gravadas as transformações (nome dado aos esquemas elaborados para integrar dados).

Após estas definições iniciais o programa está pronto a usar. Como é possível observar na Figura 3.10, a janela deste programa pode ser dividida em quatro partes.

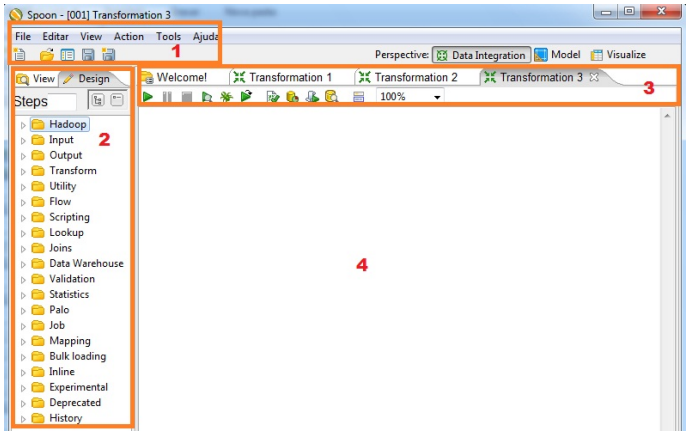


Figura 3.10: Ambiente de trabalho do Pentaho Data Integration

Na primeira parte, situam-se alguns menus com várias opções, salientando-se as seguintes:

- **File** - Gravar, abrir, novo, importar ficheiro XML ...
- **Editar** - Copiar, colar, limpar seleção, buscar meta-dados ...

- **View** - Escolha de perspetivas, zoom ...
- **Action** - Correr transformação, debug, preview ...
- **Tools** - Base de dados, assistente, opções ...
- **Help** - Dica do dia, informação sobre passo ...

Na segunda parte, encontra-se um conjunto de passos que podem ser utilizados nas transformações. Por exemplo, dependendo da origem dos dados no menu *input* é possível escolher o melhor passo para carregar esses dados. Se os dados a trabalhar estiverem numa base de dados é necessário utilizar o passo *table input*. Se por acaso os dados estiverem num ficheiro CSV então o passo utilizado já teria de ser o *CSV file input*.

Normalmente no início das transformações são utilizados sobretudo passos de *input*, uma vez que, é necessário carregar os dados a trabalhar. Para o tratamento e integração dos dados existem muitos passos que podem ser utilizados. Na Figura 3.11, pode-se observar um exemplo de transformação, onde os dados são provenientes de várias tabelas e depois através do uso do passo *Merge Join* é possível agrupar os dados. Posteriormente estes dados são enviados para um ficheiro Excel, recorrendo a passo de *output*, neste caso o *Microsoft Excel Output*.

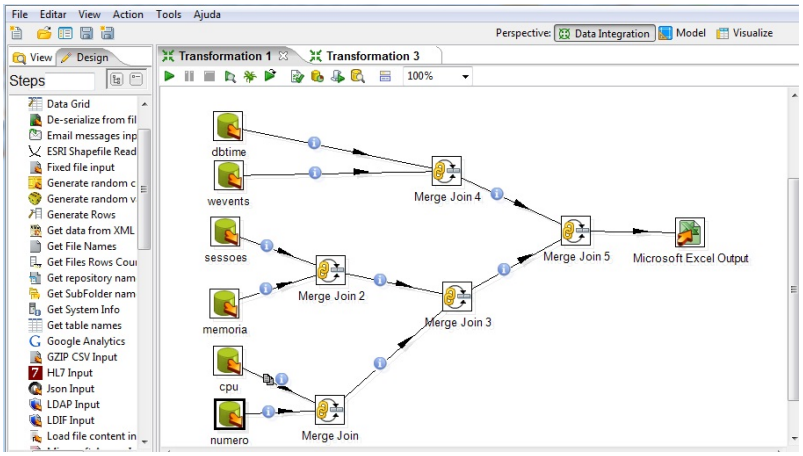


Figura 3.11: Exemplo de transformação

Na terceira parte, encontram-se os vários botões utilizados no processo de transformação. Na Figura 3.12, pode-se observar a explicação de cada um deles.

A quarta parte do ambiente de trabalho desta ferramenta, refere-se à área de trabalho. É para esta zona que se devem arrastar os passos necessários para a transformação desejada e proceder à ligação entre eles.

3.3.1 Elaboração de transformações

Nesta parte serão exemplificados os principais passos para a execução de uma transformação. Neste caso, pretende-se calcular o rácio do buffer da cache de uma base de dados.

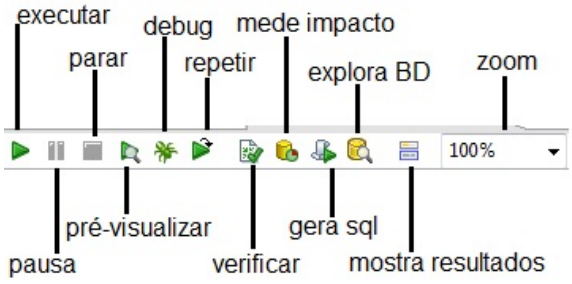


Figura 3.12: Explicação dos botões de transformação

As variáveis importantes para este rácio estão armazenadas numa tabela da base de dados. O primeiro passo é carregar essas variáveis utilizando o *table input*. Arrasta-se o passo *table input* para a zona de trabalho e fazendo duplo clique sobre ele obtendo-se um novo formulário. Neste formulário, é necessário preencher o campo relativo à base de dados, onde está armazenada a tabela, e elaborar a instrução SQL para retornar os dados da tabela pretendida. Deve-se carregar no botão pré-visualizar, para conferir se a query SQL está a retornar o que era expectável.

Seguidamente, como o cálculo do rácio buffer cache envolve várias colunas da mesma tabela deve-se adicionar o passo *Calculator*. Neste passo, efetua-se a definição os campos e a interligação entre eles. Como o rácio do buffer da cache é dado por [7]:

$$BC = 1 - (\frac{(physicalreads)}{(consistentegets + blockgets)}) \tag{3.1}$$

preenche-se o formulário de acordo com a Figura 3.13.

Por último, é importante adicionar uma forma de mostrar esta transformação. Neste caso escolheu-se um ficheiro Excel para uma melhor visualização das alterações. Para tal, adicionou-se o passo *Microsoft Excel Output*. Neste passo procede-se à definição dos campos que serão mostrados e da localização onde o ficheiro será guardado. O esquema completo desta transformação pode ser visualizado na Figura 3.14.

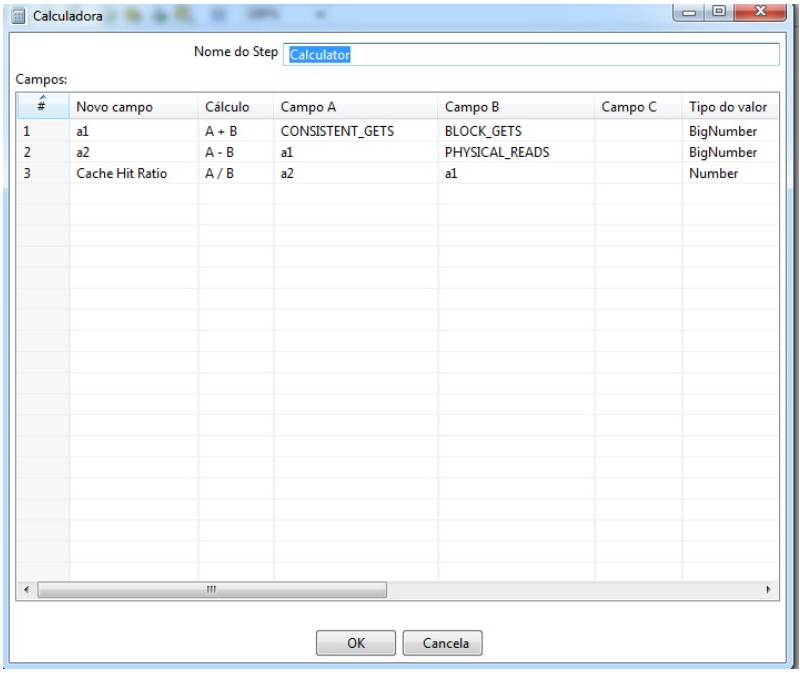


Figura 3.13: Configuração do Calculator

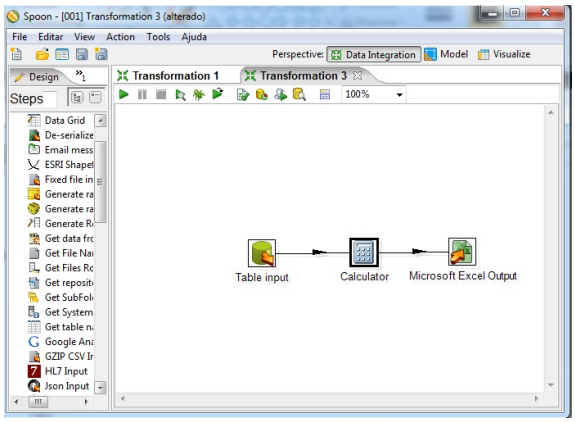


Figura 3.14: Transformação para o cálculo do rácio buffer cache

Capítulo 4

Conclusão

A ferramenta Pentaho Community é muito completa e de fácil instalação. Permite realizar com sucesso muitas das tarefas de Bussiness Intelligence que atualmente uma organização necessita.

O módulo CDE verifica-se muito versátil e apelativo no que diz respeito à criação de dashboards, mas implica alguns conhecimentos de programação para aperfeiçoar os dashboards criados.

O modulo Spoon é bastante útil para situações onde as fontes de informação são heterógeneas e quando a quantidade de dados é muito elevada.

Em suma, concluí-se que o Pentaho Community é uma solução muito versátil e na qual vale a pena ganhar um pouco de experiência.

Bibliografia

[1] S.-T. Li, L.-Y. Shue, and S.-F. Lee, "Business intelligence approach to supporting strategy-making of ISP service management," *Expert Syst. Appl.*, vol. 35, no. 3, pp. 739–754, 2008.

[2] M. Tereso and J. Bernardino, "Open source business intelligence tools for SMEs," in *Information Systems and Technologies (CISTI), 2011 6th Iberian Conference on*, 2011, pp. 1–4.

[3] C. Thomsen and T. Pedersen, "A Survey of Open Source Tools for Business Intelligence," in *Data Warehousing and Knowledge Discovery*, ser. Lecture Notes in Computer Science, A. Tjoa and J. Trujillo, Eds. Springer Berlin / Heidelberg, 2005, vol. 3589, pp. 74–84.

[4] PentahoCommunity, "Welcome to the Pentaho Community," acedido a: 15/12/2011. [Online]. Disponível em: http://community.pentaho.com

[5] Webdetails, "About | cde.webdetails.org," acedido a: 20/03/2012. [Online]. Disponível em: http://cde.webdetails.org/

[6] Jetty, "Securing Passwords - Jetty," acedido a: 29/11/2011. [Online]. Disponível em: http://docs.codehaus.org/display/JETTY/Securing+fBusinessintelligenceapproachtosupportingPasswords

[7] I. Chan, *Oracle Database Performance Tuning Guide, 10g Release 2 (10.2)*. Oracle, 2008.

Apêndice C

Scripts bash utilizados

Além do script para a recolha da percentagem do processador foram utilizados mais dois scripts. O script para a recolha da percentagem de memória da base de dados AIDA, similar ao utilizado para a recolha do processador. O código C.1 permite então a recolha do valor da percentagem de memória utilizada.

Código C.1: Script bash para recolha da utilização do processador

```
1 #!/bin/bash
2 sar -r 5 > auxcpu.txt
3 cat auxcpu.txt | tail -n1 |cut -c 27-36
```

Como é possível observar, este script só difere no parâmetro do comando *sar* seguindo o mesmo funcionamento já descrito.

Existiu a necessidade de escrever dois scripts diferentes para a base de dados SONHO, uma vez que o servidor de acesso possuí o sistema operativo HP-UNIX, que utiliza um interpretador bash diferente. Foi utilizado então o código C.2:

Código C.2: Script bash para recolha da utilização do processador no SONHO

```
1 #!/bin/sh
2 vmstat 5 3 > aux.txt
3 cat auxcpu.txt | tail -n1 | cut -c 96-98
```

É um script similar aos utilizados na AIDA com a diferença que é utilizado o comando *vmsatat*. São efetuadas de novo cinco recolhas, com três minutos de espaçamento sendo o resultado armazenado num ficheiro temporário. De seguida, utilizando o comando *cut* é possível isolar a informação pretendida, neste caso o valor de utilização do processador.

Como o comando *vmstat* fornece simultaneamente informações sobre a utilização de memória e de processador apenas foi necessário escrever o código C.3 para isolar a memória.

Código C.3: Script bash para recolha da memória livre no SONHO

```
1 #!/bin/sh
2 cat aux.txt | tail -n 1 | cut -c 29-35
```

Este script faz uso do ficheiro temporário onde são armazenadas as informações provenientes do comando *vmstat*, isolando neste caso informações relativas à memória. É importante salientar que as informações relacionadas com a memória obtidas com o *vmstat*, não são dadas em percentagem mas sim em quantidade de memória livre.

Apêndice D

Especificação das Tabelas

Os dados provenientes da monitorização, bem como os dados relativos às anormalidades detetadas e possíveis falhas encontram-se armazenados num conjunto de tabelas numa base de dados distinta das que estão a ser monitorizadas. Foram criadas as seguintes tabelas:

- r_registo_aida, r_registo_aida2, r_registo_sonho - para o armazenamento, em cada minuto, dos valores das métricas escolhidas provenientes dos dois nodos da base de dados AIDA e da base de dados SONHO.
- normal_stat_aida, normal_stat_aida2, normal_sat_sonho - contém informação sobres os limites de normalidade e percentis das bases de dados avaliadas.
- aida_anormal, aida_anormal2, sonho_anormal - armazena informação relativa às situações anormais detetadas. Quando o valor recolhido ultrapassa os limites pré-definidos é registada aqui bem como o score de gravidade a ela associado.
- r_falha, r_falha2, sonho_falha - para o registo de possíveis falhas. Quando o programa de monitorização é interrompido é registada a data e a hora dessa interrupção bem como a descrição da causa que a originou.

Como a estrutura das tabelas é similar apenas será apresentado o código elaborado para a criação das tabelas relativas ao nodo 1 da base de dados AIDA.

Código D.1: SQL para criação de tabelas

```
1
2 create table r_registo_aida(
3 datasistema date,
4 name varchar2(120),
5 value number(20,5));
```

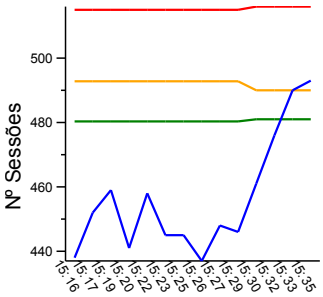
```
6
7  create table normal_stat_aida(
8  datasistema varchar2(5),
9  name varchar2(25),
10 p25 number (15,5),
11 p75 number (15,5),
12 p80 number (15,5),
13 p90 number (15,5));
14
15 create table aida_anormal(
16 datasistema date,
17 name varchar2(120),
18 value number(15,5),
19 score number (1));
20
21 create table r_falha(
22 datasistema date,
23 name varchar2(150));
```

Apêndice E

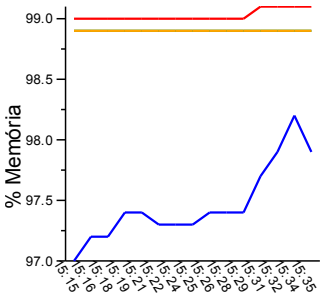
Excertos dos dashboards

Monitorização:chp-ora01

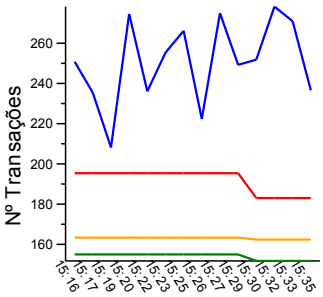
Nº Sessões



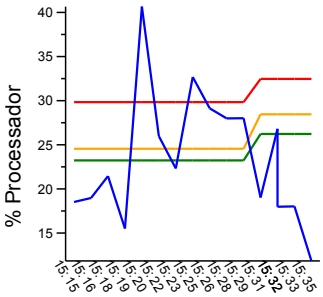
Memória



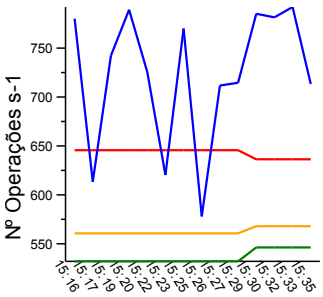
Transacções



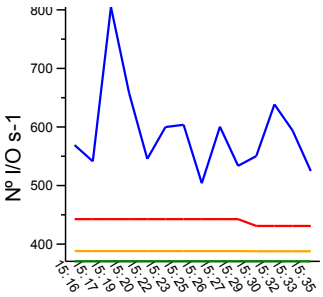
Processador



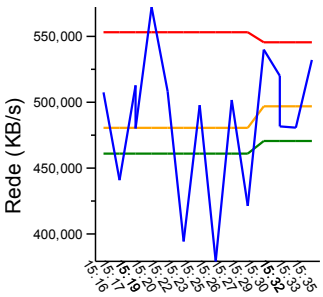
Nº Operações



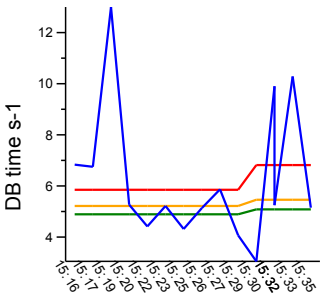
Input/Output



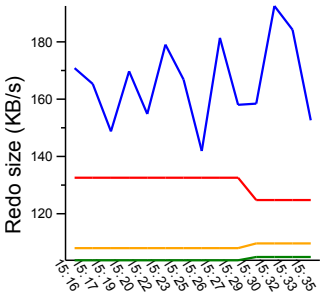
Tráfego de Rede



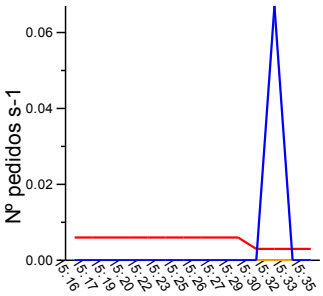
DB time



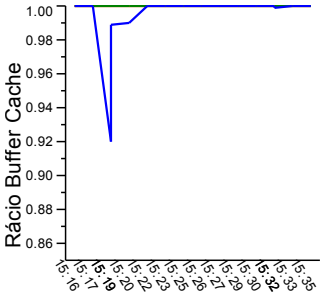
Redo Size



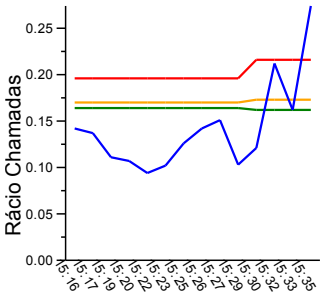
Ped. Espaço Redo



Buffer Cache



Chamadas Recursivas



Monitorização AIDA: Geral

Monitor Geral (chp-ora01)

Search:

Hora	Estatística	LSC	Valor	Diferença
16:30	Percentagem de utilização do processador	20.935	15.56	
16:30	Percentagem de utilização da memória	98.7	97.3	
16:30	Nº de transações por segundo	154.35	188.16667	
16:30	Pedidos de espaço para redo log por segundo	0	0	
16:30	Rácio Chamadas Recursivas	0.15866	0.08191	
16:30	Quantidade de entradas redo (kb/s)	103.531235	119.12129	
16:30	Rácio Buffer Cache	1	1	
16:30	Nº de operações por segundo	510.05	488.58333	
16:30	Nº de pedidos I/O por segundo	363.746115	439	
16:30	Nº de sessões	458	440	
16:30	DB time por segundo	4.5935	2.65	
16:30	Percentagem de utilização da memória	98.7	97.3	

Showing 1 to 12 of 12 entries

Monitor Geral (chp-ora02)

Search:

Hora	Estatística	LSC	Valor	Diferença
16:45	Percentagem de utilização da memória	98.1	97.6	
16:45	Nº de transações por segundo	138.079165	134.8	
16:45	Percentagem de utilização do processador	23.6	18.08	
16:45	Percentagem de utilização da memória	98.1	97.6	
16:45	DB time por segundo	3.95	2.71667	
16:45	Nº de sessões	443	411	
16:45	Nº de pedidos I/O por segundo	329.55	359.7	
16:45	Nº de operações por segundo	455.410645	427.35	
16:45	Rácio Buffer Cache	1	1	
16:45	Quantidade de entradas redo (kb/s)	94.27816	90.76471	
16:45	Rácio Chamadas Recursivas	0.156645	0.15006	
16:45	Pedidos de espaço para redo log por segundo	0	0	

Showing 1 to 12 of 12 entries

Gráfico Pormenorizado

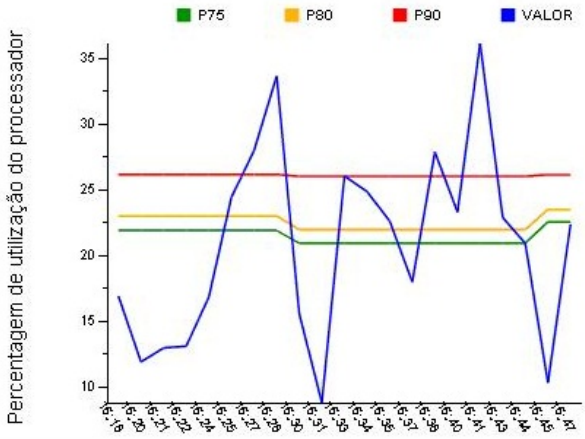
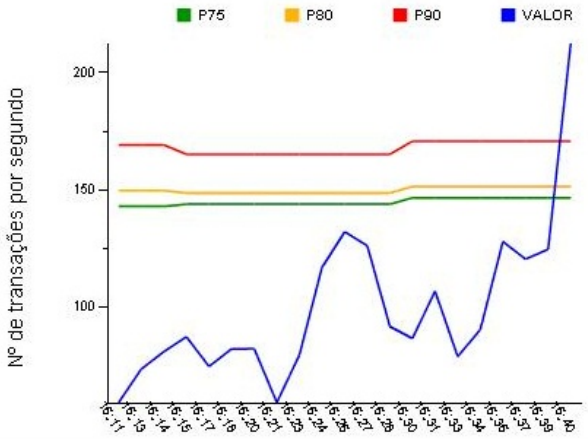


Gráfico Pormenorizado

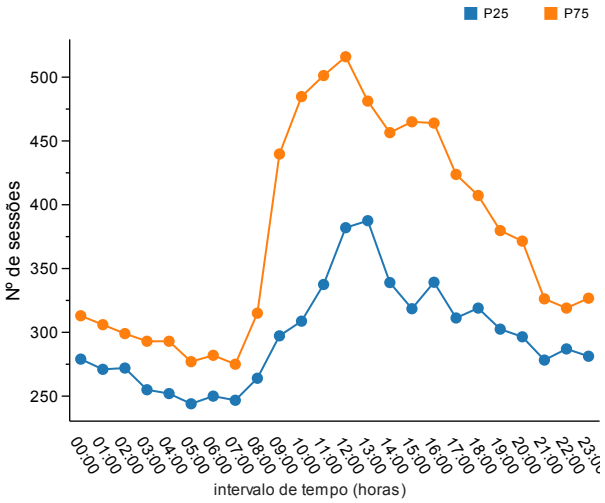


Análise AIDA: Workload normal

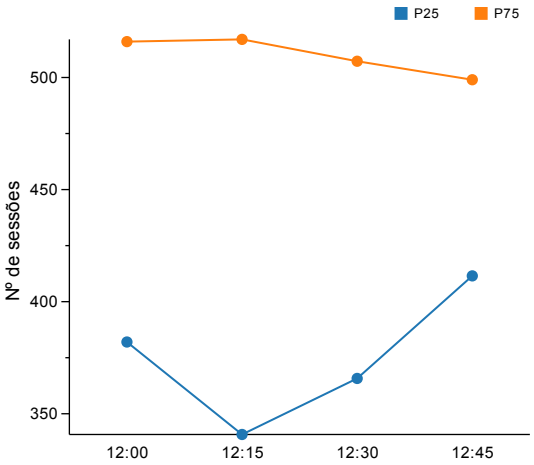
Escolha por favor

Nº de sessões

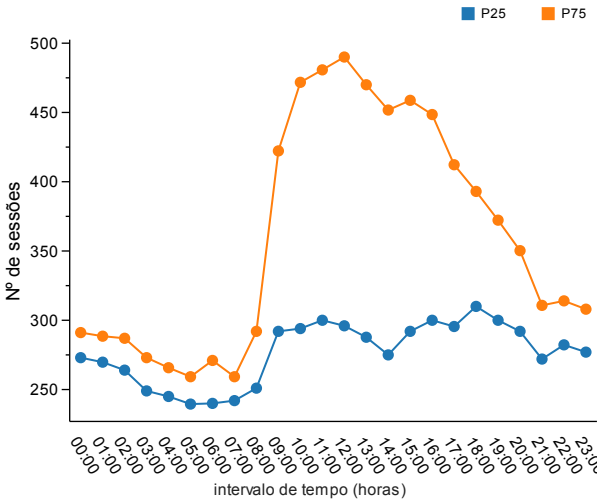
chp-ora01-vip



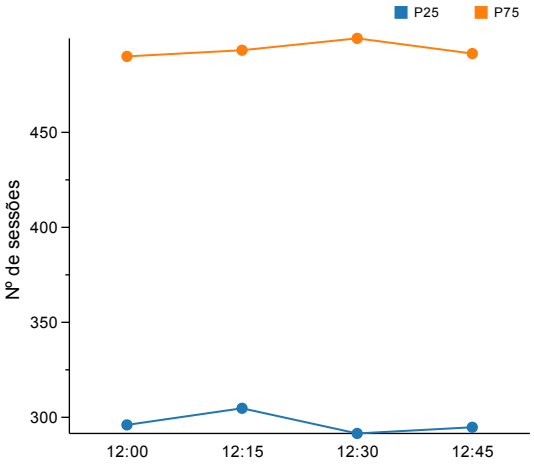
Det. chp-ora01



chp-ora02



Det. chp-ora02



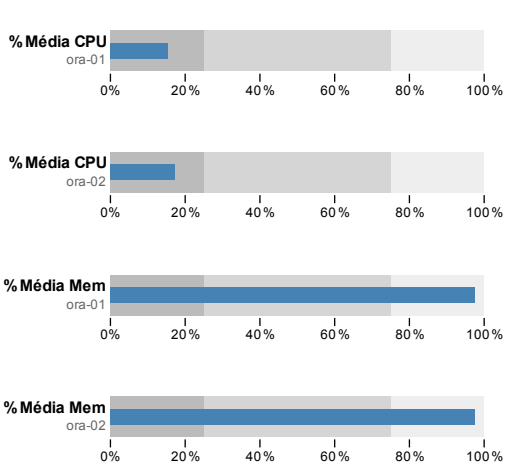
Resumo Workload

Show 10 entries Search:

Nome	Média (nodo1)	Média (nodo2)	Total (nodo1+nodo2)	Diff (nodo1-nodo2)
Nº de transações por segundo	113.108	102.108	215.216	
DB time por segundo	3.403	3.011	6.414	
Nº de sessões	344.598	329.859	674.457	
Nº de pedidos I/O por segundo	331.731	297.753	629.484	
Nº de operações por segundo	382.593	352.907	735.5	
Quantidade de entradas redo (kb/s)	70.201	79.548	149.749	
Pedidos de espaço para redo log por segundo	0.595	0.004	0.599	
Volume tráfego de rede por segundo (bytes/s)	354656.658	324434.321	679090.979	

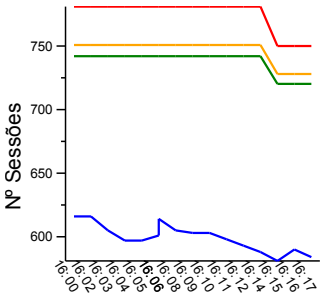
Showing 1 to 8 of 8 entries

Processador e Memória

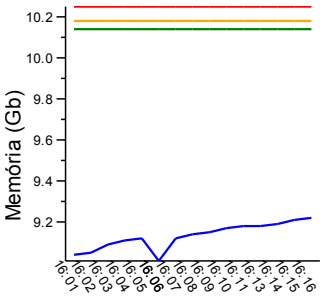


Monitorização:chp-sonho

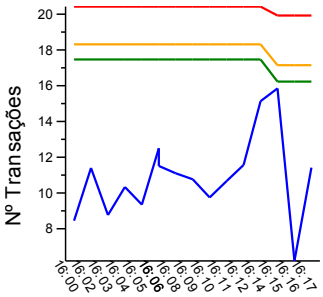
Nº Sessões



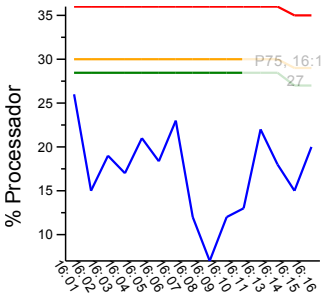
Memória



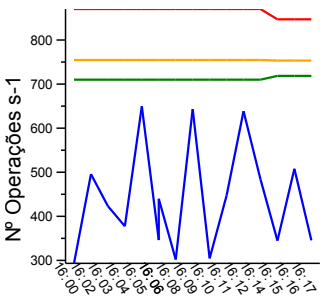
Transacções



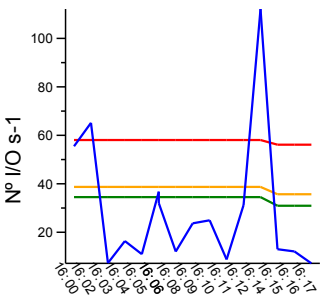
Processador



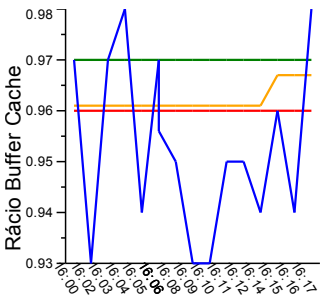
Nº Operações



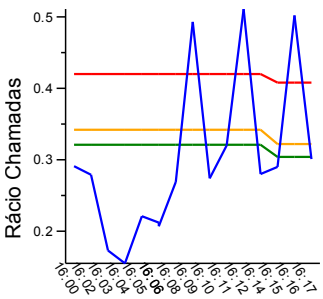
Input/Output



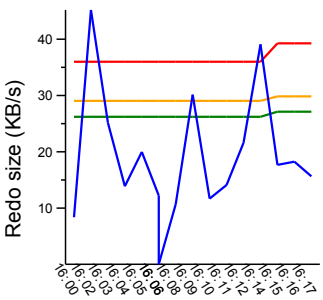
Buffer Cache



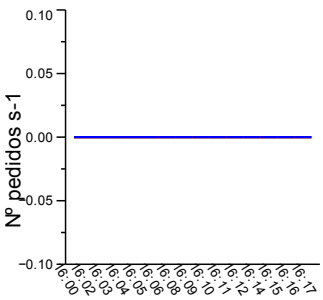
Chamadas Recursivas



Redo Size



Ped. Espaço Redo

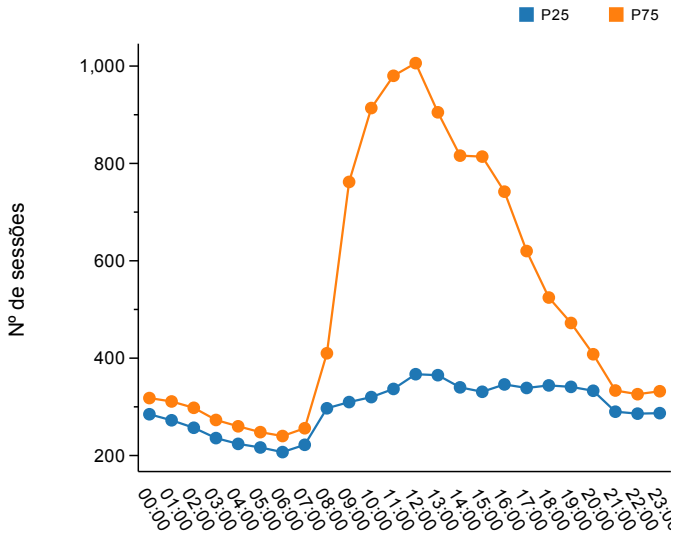


Análise SONHO: Workload normal

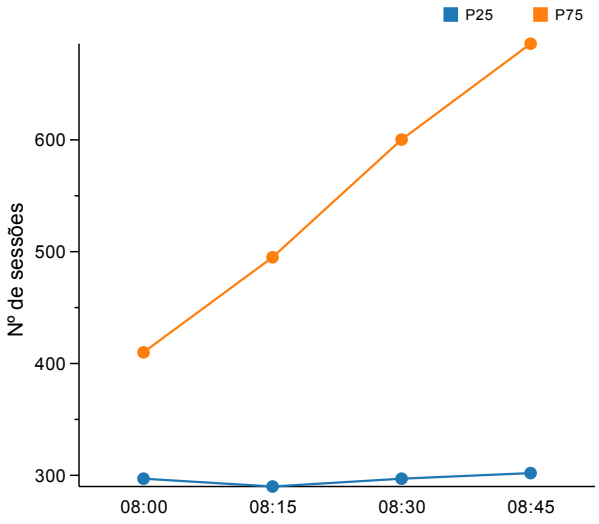
Escolha por favor

Nº de sessões

chp-ora01-vip



Det. chp-ora01



Resumo Workload

Show 10 entries Search:

Nome	Média	Desv. Padrão
Nº de transações por segundo	10.244	8.632
Percentagem de utilização do processador	26.522	7.228
Memória Livre	2685710.462	24679.414
Nº de sessões	524.994	265.972
Nº de pedidos I/O por segundo	35.667	43.684
Nº de operações por segundo	748.283	1221.085
Rácio Buffer Cache	0.961	0.006
Quantidade de entradas redo (kb/s)	36.076	55.791
Rácio Chamadas Recursivas	0.384	0.172
Pedidos de espaço para redo log por segundo	0	0.002

Showing 1 to 10 of 10 entries

Apêndice F

Artigos

Step Towards Fault Forecasting in Hospital Information Systems

Paulo Silva*, Cesar Quintas†, Manuel Santos‡, Antonio Abelha§ and Jose Machado§
§Universidade do Minho, Computer Science and Technology Center - CCTC, Braga, Portugal
†Centro Hospitalar do Porto, Porto, Portugal
‡Algoritmi Center, Guimares, Portugal

***Abstract*—Nowadays, many organizations consider databases indispensable tools for their daily tasks. Particularly in healthcare units, databases have a vital role, since they archive very important information about patients' clinical status, therefore, it is crucial that databases are available twenty-four hours a day, seven days per week. Healthcare units have already implemented fault tolerant systems, which intended to ensure the availability, reliability and disaster recovery of data. However, these mechanisms do not allow to take preventive actions in order to avoid the occurrence of faults. In this context, the necessity of the development of faults prevention and prediction systems is emerging. These systems can predict faults with some time in advance and provide taking early action to solve problems. The objectives of this paper are: monitor database performance and adapt a forecasting model used in medicine (Modified Early Warning Score - MEWS) to database reality.**

***Index Terms*—Medical Informatics, Hospital Information Systems, Database Monitoring.**

I. Introduction

Databases are powerful tools to record and manage of the large amount of organizations' data. They have several components since software components (application, management) until hardware components (disks, memory, processor, connectors) that are coordinated by the Database Management System (DBMS). Besides these components the DMBS has to take into account the users and their satisfaction, to perform their requests in time [1][2][3].

Actually, DBMS solutions provide by Oracle are one of the most well know and most used solutions for database management. Over the years, organizations have increased the use of databases and DBMS, which, for simplicity, sometimes are all denominated only by database [3]. Today, many organizations consider databases indispensable tools for daily tasks [4]. Particularly in healthcare units, databases have a vital role, since they archive very important information about patients' clinical status and other information relevant to proper functioning of the healthcare unit. So it is crucial that these databases provide a high level of security to ensure permanent data integrity and availability. One of the fundamental characteristics of database security is the availability. In these units, it is very important that databases are available twenty-four hours a day, seven days per week. Therefore, database availability is a complex and crucial feature [3].

Database availability is a complex question, because the fact of the database is "up" is not a synonym that it is available. There are several problems that influence database availability, such as: database can be inaccessible due to network problems or a virus that bar user's connections; the database can be too slowly and therefore the users' requests are not satisfied. The slowness of database can be related to resource limitation or an elevate number of operations. Database has intermittent faults and loose the user's confidence, because of its unavailability [5][6]. Database faults can happen frequently, and can be of several types depending of the vendor and environment where database is installed. Nevertheless, it is very important that these faults are transparent for users, so that database continues available despite faults [7]. Healthcare units have already implemented some fault tolerant systems, which intend to ensure the availability of data when a fault occurs. However, it is very important the development of fault's forecasting and prevention systems, which allow to taking early actions to solve future problems by identified the abnormal situations. An alternative to these mechanisms can be the forecasting models that have been used in critical areas such as medicine [8].

The objective of this paper is the development of a model for forecasting and prevention of database faults by adapting an existent forecasting model in medicine (Modified Early Warning Score - MEWS) [8]. A monitoring program was developed and with business intelligence tools knowledge was extracted in order to the study of the data. This is a base for the development of a forecasting fault application.

II. Healthcare Information Systems

Over the years, healthcare units have increased the utilization of computer systems in many of their services in order to introduce new methodologies for problem solving [9][10]. In this sense, healthcare information systems have emerged. These systems are responsible for optimizing the set of information which exists in a healthcare unit. They proceed to the collection, processing, and data management of all stakeholders (patients, nurses, doctors) and of all services of healthcare unit. Thus, in a particular situation, the authorized people have a fast and efficient access to relevant information [9]. Rapidly the information systems have taken a key role in healthcare units. Since 1990s, hospital information systems, in Portugal was implemented the Hospital Information Integrated System (SONHO). This system had as objectives merge clinical and administrative information. The evolution

of these systems is the genesis of the creation of the electronic medical record, and of the addiction of two modules: the medical support system (SAM) related to medicine and the nurse support system (SAPE), related with nursing care [11].

However, in healthcare units there is a lot of other large information systems, they are distributed and heterogeneous systems and in order to communicate an effort was made to develop interoperability [12]. To solve this problem a dynamic framework (AIDA- Agency for the Integration, Diffusion and Archive) was developed. This framework is composed of pro-active agents that are responsible for promoting communication by sending/receiving information, and managing and saving the information. Thus, AIDA shares information and knowledge among every information systems, namely: the administrative information system (AIS); the medical support information system (MIS); the nursing support information system (NIS); the EMR information system; the department information systems (DIS) of all the departments or services [12][13]. The information is archived in very large databases which must be available every day of the year because their information is vital for solving the patients' problems and for hospital management. Therefore, it is crucial to ensure the integrity and permanent availability of data in case of faults [4]. To achieve this goal it is necessary to use tolerance fault mechanisms, which determines data redundancy, component redundancy or both [3]. Through this process of redundancy, it is possible for a database to still be available in spite of failures in the hardware or software components of the system. Some mechanisms also allow for load balancing and recovery in extreme situations, such as fire [4][14].

In the particular case of the AIDA and SONHO databases of Centro Hospitalar do Porto are based in an Oracle Real Application Cluster system (RAC). This mechanism is provided by Oracle for improving the availability and scalability of databases. This goal is achieved through architectures presented in Figures 1 and 2. This architecture is composed by a shared database which can be accessed through the server/computer that contains a database instance and an Automatic Storage Management (ASM) instance [3][15][16]. In addition to the RAC system there is one data guard solution (Figure 3). A data guard solution consists in one or more standby databases (replicas of the original database) which should be in different places. Thus, if for some reason the main database is unavailable, the system will still work by using one of the standby databases. It is important that the main and the standby databases are synchronized and the access is read-only during recovering [4].

In spite of all advantages, the fault tolerance tools do not allow the focus on the faults themselves, but only on the faults' effects, trying to minimize them. Thus, it is necessary to monitor the components of the database, trying to understand their faults and symptoms.

The AIDA and SONHO databases are large databases which a large quantity of information: AIDA includes around 5451 tables, SONHO has around 2255 tables and with a large number of sessions and processes that run permanently.

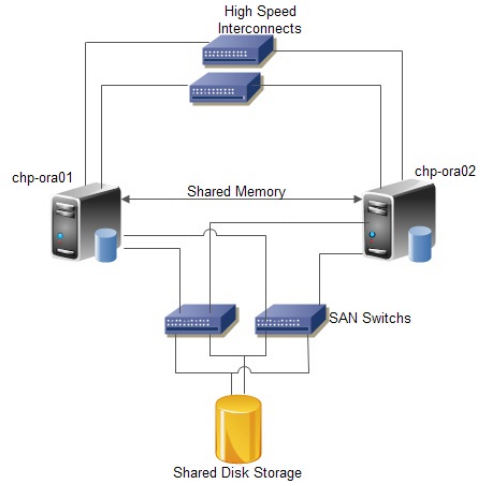


Fig. 1. AIDA RAC Architecture

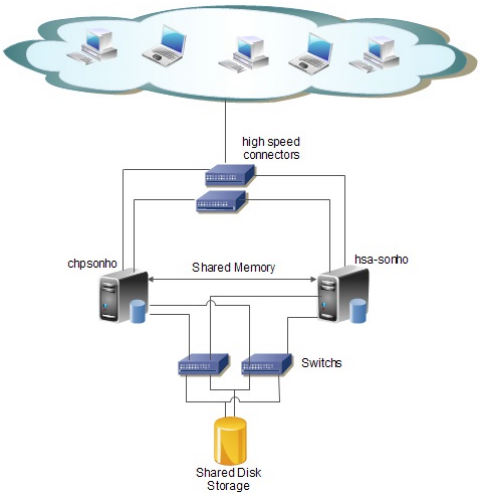


Fig. 2. SONHO RAC Architecture

Therefore, most often database faults are related to resource database limitation. So, the focus of this paper is in forecasting fault database related with database resources, adapting the forecasting model used in Medicine (MEWS) of the world of databases.

III. MEWS

In medicine a model for the prediction, in advance, of serious problems of health is already used. This model is Modified Early Warning Score (MEWS) and it assumes that a serious problem of health is, sometimes, preceded by physiological deterioration. So it is an important procedure to monitoring patient's vital signs that depends on the deviation from normal assigned scores. In Table I, it is possible to find the vital signals and the score associated to each set of values.

These scores are added and it is determining the level of risk of each patient, trying understanding when a serious problem occurs [8][17]. This model is very important because the resources of intensive care units are limited, and then it is

TABLE I
MEWS SCORES

MEWS SCORE	3	2	1	0	1	2	3
Temperature (C)		< 35.0	35.1-36.0	36.1-38.0	38.1-38.5	> 38.6	
Heart rate (min1)		< 40	41-50	51-100	101-110	111-130	> 131
Systolic BP (mmHg)	< 70	71-80	81 - 100	101 - 199		> 200	
Respiratory rate (min1)		< 8		8-14	15-20	21-29	> 30
SPO2	< 85	85-89	90-93	> 94			
Urine output (ml/kg/h)	Nil	< 0.5					
Neurological		New confusion/agitation		Alert	Reacting to voice	Reacting to pain	Unresponsive

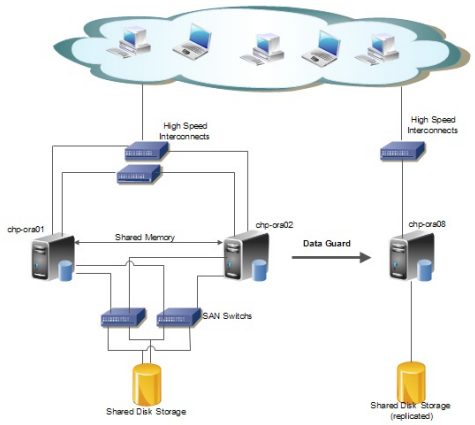


Fig. 3. AIDA Complete Architecture

possible to identify patients at risk [17]. The concept of Early Warning Score was been introduced in 1997 and the vital measured signs were: temperature, systolic blood pressure, respiratory rate, heart rate and neurological activity [17].

However, some years later several studies have concluded that more parameters for the success of this model were needed. So, in 1999 a new model appeared with the use of two additional variants: urine output and saturation of hemoglobin with Oxygen (SPO2) [17][18][19]. The monitoring of patient's vital signs should be continuous and all values must be archived to understand the behavior of the vital signs over time [18][20].

Normally, if any of the parameters have the score equal to two, the patient must be in observation. In the case of the sum of scores is equal to four, or there is an increase of two values the patient requires urgent medical attention. In a more extreme situation, if a patient has a score higger than 4, is life is in risk [8][20].

The use of this model has a lot of advantages in the process of preventing patient's medical complications, such as:

- Allow defining priority for the medical interventions [17].
- Improve the monitoring and observation of the patients since it gives an indication of physiological parameters [20].
- Support the medical decision, it is more objective because it is based on quantitative criteria [18].

IV. METHODOLOGY

A. Monitoring database performance

The first step to construct a forecasting database's fault system is monitoring database components that can cause faults. The monitoring database components is not only for fault diagnostic but also for improvement of security because with database monitoring it is possible, for example, to identify an abnormal resource utilization for an unknown user. Therefore, the use of monitoring tools has increase in organizations [21]. The monitoring difficulty increases with the dimension and complexity of databases. Normally, hospital's database had several components and complex architectures. Therefore, hospital database monitoring is not a simple task; however, the Oracle systems provide several tools to help in this process. A sample of these tools is the performance views (v$). These are a set of views where useful information for monitoring process is available. A view looks like a simple database table, but since views show the actual system state, it is impossible do editing operations, is only possible to consult information. These views are sometimes called dynamic performance views because their content is refreshed several times during a database instances' life [22][23].

The most important views related to this paper are the ones which contain information about database component statistics, such as processor, memory, disk, sessions and users. These statistics can be divided into three main groups [22][24]:

- **Time Model Statistics** - Provides information about the time spent in the database calls.
- **Wait Event Statistics** - Provides information about the time the server process/thread had to wait for an event to complete before being able to continue processing.
- **Session and System Statistics** - Provides information about the use of system resources, about users and their sessions.

However, this tool is not enough. It is necessary to use others tools more related with physical resources. Usually commands of operating system, which in this case are UNIX command, are used. There are several commands that can help in the monitoring process. For example, we can use the command **top** or the command **sar** to monitor processor (CPU) and memory usage, and we can use the command **ps** to see the active process in the moment. Using these tools together we can do a more efficient monitoring [25].

There are others tools for database monitoring such as Automatic Workload Repository (AWR), or Nagios [22]. However, it is chosen the performance views for two reasons: it provides a easy connection with the Business intelligence tool (Pentaho) for presentation and data analysis; and it is easier to define the zone of monitoring thus reducing the cost associated to monitoring.

B. Business Intelligence(BI) tool

In this paper, the use of BI tools is useful because database monitoring provides a large quantity of data which are difficult to interpret and manipulate.

The term Business Intelligence (BI) was introduced by Howard Dresner of the Gartner Group to describe a set of concepts/methods which utilize new technologies to improve the business on decision making [26][27]. Over the years, many different BI definitions have appeared depending on the approach (managerial, technical, system-enabler) followed by each author [26].

In general, BI is related to a set of concepts, methods, processes, and tools used to improve business, particularly, decision-making. To achieve this goal, several steps are necessary. First, it is necessary to acquire and integrate data from different sources and archives. This data are stored in a large database (data warehouse). Then it becomes possible to perform several kinds of analyses to obtain useful information for decision making and reasoning for past experiences which allows for a precise understanding of business dynamics [26][28][29]. Implementation of these techniques is made by several analysis tools, such as production reporting, end-user query, data mining, dashboard, On-Line Analysis Processing (OLAP) and planning tools [26]. The utilization of BI in organizations brings many benefits, such as cost-saving in data consolidation, time savings for user requests, more and better information, better decisions, and more support for the accomplishment of strategic business objectives [28].

After reviewing several studies about BI open-source tools, it was observed that Pentaho Community version is more appropriate to accomplish this work's objectives [30][31]. The Pentaho tool was released in 2004 having two versions: a commercial version (Enterprise Edition) and a free version (Community Edition). The community version provides several features, such as reporting, dashboards, charts, data mining tools and data exportation. This tool consists of several java applications which means that it can be used on different computational platforms. There is a main application, the bi-platform where applications can be integrated in order to turn available additional functionalities [30][32]. The mentioned applications are [32]:

- **CDF-CDE** - Allows the user to develop dashboards, in order to show more clearly the data collected/treated.
- **Pentaho Data Integration (Spoon)** - Delivers a powerful Extraction, Transformation and Loading (ETL) capabilities.
- **Mondrian** - Allows business users to analyze large quantities of data in real-time (OLAP).
- **Pentaho Report Designer** - Allows the user to create relational and analytical reports from a wide range of data-sources, in several output files.
- **Weka** - Used to apply machine learning and data mining techniques. Weka is an independent free program which can be integrated with Pentaho by Weka Scoring Plugin, in the Pentaho Data Integration application.

In this paper, it is only used Pentaho Data Integration for manipulation of data and Community Dashboard Editor for developing charts and dashboards.

C. Explain of the statistics

Several statistics can be used for determining normal database performance. We developed a forecasting fault model for the database workload. The chosen statistics are related with the load of databases. The chosen statistics are: DB time, number of transactions, number of executions, calls ratio, number of current logons, processor and memory utilization, size of redo, buffer cache ratio, amount of I/O requests, amount of redo space requests and network traffic [4][24][33].

- **DB time** - This statistic gives information about database response time. The response time is the period between an initial user request until the return results. This time should be minimized. In Oracle systems, this time is a sum of total time (include CPU time, IO time, Wait time) spent on all requests from users. Therefore it is a good indicator of the workload of the system as it usually increases with the number of users and number of applications, and sometimes can also increase due to the larger transaction, or when there is some problem in the system [4][34][35].
- **Number of transactions** - In databases, transactions are considered units of work. In Oracle databases, the number of transactions can be obtained by summing the values of statistics "user commits" and "user rollbacks" because each transaction always ended with a command "commit" and any undo operation as a command "rollback" [4][5][25].
- **Number of executions** - One transaction may trigger a set of operations in database depending of query and information needs. Therefore, it is important to collect information about the number of operations. In Oracle databases this information can be obtained collect, the "execute count" statistic [25].
- **Calls ratio** - One transaction results in several calls. These calls can be of two types: user calls or recursive calls. When, after a request from a user, only one SQL statement runs is considered an user call. If the statement SQL requested by the user, need another SQL statement to meet the request of the same, then there is a recursive call. This ratio can be calculated using statistics "user calls" and "recursive calls". Ideally, this ratio is as low as possible, because a large number of recursive calls may indicate: problems with the design of tables, an increase of sorts because of disk cache memory problems, or excessive running trigers [23][36].

- **Number of current logons** - The peaks of users are not directly related to the users, but with the number of sessions that may be simultaneously using the database, since a user can have more than one session. This value can be obtained through the statistical "logins current". It is important to determine the time of day, that have a higher number of sessions since each session is assigned to a portion of memory (PGA), which can influence the database performance [24].
- **Processor utilization** - It is important to analyze the amount of processor (CPU) used, because if the processor is in little-used may indicate that there is a high level drive I/O, which may indicate a problem. Other behavior indicative of problems, is the case in where an CPU is widely used but only for a small number of programs, which means that some processes never never have access to the processor. The observation of the host CPU, in percentage, can be obtained by operating system commands [24].
- **Memory utilization** - The memory is a key component to the speed of the database systems, since, if the data stays in memory, it improves the response time. However, if memory is busy, response time tends to decrease. Therefore it is important to determine when there are peaks of memory usage since they can lead to a problem of performance of the system [5].
- **Size of redo file** - Represents the amount of redo entries (kb). It is very important due to the function of redo files (archive the modifications in database). An increase of this statistic value can indicate an increase of the number of operations, therefore, the database load also increases [23].
- **Buffer cache ratio** - This ratio show the percentage of data that is in memory cache, rather than disk. This ratio can be calculate using the equation:

$$BC = \frac{1 - (physicalreads)}{(consistentegets + blockgets)} \quad (1)$$

  However, care must be taken because if is used cumulative values, the ratio does not show any changes and there may be some error, although it can not be seen because the variables' magnitude [5][35].
- **Amount of I/O requests** - The I/O operations demand a lot of time, and an excess of this kind of operations can indicate problems in memory. If there is no free space in memory, it is necessary to put data on the disk, which can damage the performance of the database [24].
- **Amount of redo space requests** - When there is no space in Redo log buffer more space is requested to put redo entries. This can happen due to small size of redo log buffer or due to an abnormal load of database that increases the number of operations [36].
- **Volume of network traffic** - The several database components are often connected through network and the user requests come from network. Therefore, the network is also important for database performance. If a volume of network is increasing rapidly, the database can be slow and compromise users' requests, so it is a very important statistic [22].

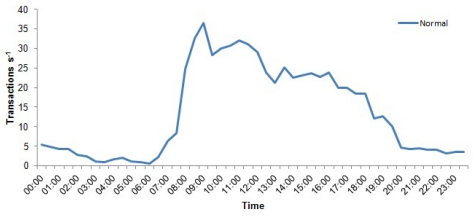


Fig. 4. Transactions per second in SONHO

D. Process

Initially, a java application was developed to collect data related with statistics described above from the two major databases (AIDA, SONHO) of Centro Hopitalar do Porto. The application collected information during a week. With this information it was possible to make a model that represents a normal behavior of workload database. A number of charts and dashboards were created, providing important conclusions about the performance, normal behavior and utilization peaks of the database system. Using percentiles, some limits were defined to represent intervals of normal behavior. Thus, it is possible to detect abnormal situations as those statistics on the measured value is not within limits.

A new program was developed for catching the abnormal situations. Depending on the value of the deviation and the frequency of times that happen abnormal situations are assigned granted scores like it is done in MEWS. According to these scores, alerts will be sent to database administrator. This program is upgradeable because it calculates new limits when a new collect has done, using values within the limits, so the program continuously learn.

V. Results

This program is based in learning rules. Dynamically it creates a new model from normal behavior. It is presented an example for one database. In Figure 4, it is represented a limit of a normal number of transactions per second in SONHO database along the day is represented.

It is possible to see that the database has more transactions activity during morning, due to higher number of users connected. In Figure 5, abnormal situations that occur along one day are represented. The abnormal situations occur at the end of the morning. There is a repetition of abnormal values, this is a serious situation, so the score is 3. A warning of possible fault should be issued. However, database continues in operation despite the alerts.

VI. Conclusions

This paper shows how important is the theme of faults forecasting in healthcare databases. Despite being a complex theme, we propose a model to represent the normal workload for databases AIDA and SONHO. With this model, it

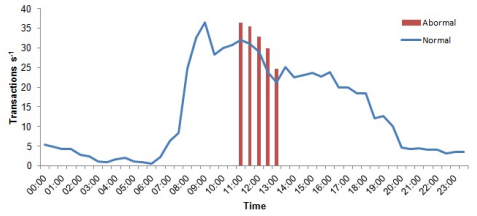


Fig. 5. Abnormal situations: Transactions per second in SONHO

is possible to do several analysis and with help of other mathematical tools it is possible to develop a program that detects the abnormal situations. According to the frequency and intensity of these abnormal situations it is possible to create a decision table with scores that represent the situation gravity. The scores represent the situation gravity. Thus, the MEWS were adapt to database reality through an upgradeable and learning forecasting fault system.

VII. Acknowledgment

We would like to thank the Centro Hospital do Porto, in Oporto for having made available their database systems to develop this project. This work is funded by National Funds through the FCT - Portuguese Foundation for Science and Technology within project PEst-OE/EEI/UI0752/2011.

References

[1] T. Connolly and C. Begg, *Database Systems A Pratical Approach to Design, Implementation, and Management*, 4th ed. Harlow: Addison-Wesley, 2005.

[2] G. W. Hansen and J. V. Hansen, *Database Management and Design*, 2nd ed. Upper Saddle River, N.J.: Prentice Hall, 1996.

[3] S. Drake, W. Hu, D. McInnis, M. Skold, A. Srivastava, L. Thalmann, M. Tikkanen, O. y. Torbjornsen, and A. Wolski, "Architecture of Highly Available Databases," in *Service Availability*, ser. Lecture Notes in Computer Science, M. Malek, M. Reitenspie, and J. Kaiser, Eds. Springer Berlin / Heidelberg, 2005, vol. 3335, pp. 1–16. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-30225-4_1

[4] R. Godinho, "Availability, Reliability and Scalability in Database Architecture," MSc Thesis, Universidade do Minho, 2011.

[5] R. Schumacher, *Oracle Performance Troubleshooting With Dictionary Internals SQL & Tuning Scripts*. Kittrell: Rampant TechPress, 2003.

[6] F. Piedad, *High availability : design, techniques, and processes*. Upper Saddle River: Prentice Hall, 2001.

[7] O. Yi, "Highly Available Database Systems," in *Integrated information systems and databases seminar*, 2006.

[8] C. P. Subbe, M. Kruger, P. Rutherford, and L. Gemmel, "Validation of a modified Early Warning Score in medical admissions," *QJM*, vol. 94, no. 10, pp. 521–526, 2001. [Online]. Available: http://qjmed.oxfordjournals.org/content/94/10/521.abstract

[9] L. Mota, "Nursing information systems: a study on the relevance of information." MSc Thesis (in Portuguese), Universidade do Porto, 2010.

[10] E. Ammenwerth, J. Brender, P. Nykanen, H.-u. Prokosch, M. Rigby, and J. Talmon, "Visions and strategies to improve evaluation of health information systems Reflections and lessons based on the HIS-EVAL workshop in Innsbruck," *International Journal of Medical Informatics*, 2004.

[11] S. Lameirao, "Hospital management and use of information systems: application to CHVR-PR," MSc Thesis (in portuguese), Universidade de Tras-os-Montes e Alto Douro, 2007.

[12] J. Machado, V. Alves, A. Abelha, and J. Neves, "Ambient intelligence via multiagent systems in the medical arena," *Engineering Intelligent Systems for Electrical Engineering and Communications*, vol. 15, no. 3, pp. 151–157, 2007.

[13] J. Machado, A. Abelha, P. Novais, J. Neves, and J. a. Neves, "Quality of service in healthcare units," *Int. J. Computer Aided Engineering and Technology*, vol. 2, no. 4, pp. 436–439, 2010.

[14] M. P. Sullivan, "System Support for Software Fault Tolerance in Highly Available Database Management Systems," Tech. Rep., 1992.

[15] M. Cyran, P. Lane, and J. Polk, *Oracle Database Concepts, 10g Release 2 (10.2)*. Oracle, 2005. [Online]. Available: http://docs.oracle.com/cd/B19306_01/server.102/b14220.pdf

[16] M. Bauer and R. Strohm, *Oracle Real Application Clusters Administration and Deployment Guide, 11g Release 1 (11.1)*. Oracle, 2009. [Online]. Available: http://docs.oracle.com/cd/B28359_01/rac.111/b28254.pdf

[17] A. Albino and V. Jacinto, "Implementation of early warning score - EWS," Centro Hospitalar do Barlavento Algarvio, EPE, Portimão (in portuguese), Tech. Rep., 2009.

[18] J. Gardner-Thorpe, N. Love, J. Wrightson, S. Walsh, and N. Keeling, "The value of Modified Early Warning Score (MEWS) in surgical in-patients: a prospective observational study." *Annals of the Royal College of Surgeons of England*, vol. 88, no. 6, pp. 571–5, Oct. 2006.

[19] C. P. Subbe, R. G. Davies, E. Williams, P. Rutherford, and L. Gemmell, "Effect of introducing the Modified Early Warning score on clinical outcomes, cardio-pulmonary arrests and intensive care utilisation in acute medical admissions*," *Anaesthesia*, vol. 58, no. 8, pp. 797–802, 2003. [Online]. Available: http://dx.doi.org/10.1046/j.1365-2044.2003.03258.x

[20] G. Devaney and W. Lead, "Guideline for the use of the modified early warning score (MEWS)," Outer North East London Community Services, Tech. Rep., 2011.

[21] S. Nair, "The Art of Database Monitoring," *Information Systems Control Journal*, no. Ccm, pp. 1–4, 2008.

[22] I. Chan, *Oracle Database 2 Day + Performance Tuning Guide, 10g Release 2 (10.2)*. Oracle, 2010.

[23] K. Rich, *Oracle Database Reference, 10g Release 2 (10.2)*. Oracle, 2009.

[24] I. Chan, *Oracle Database Performance Tuning Guide, 10g Release 2 (10.2)*. Oracle, 2008.

[25] C. Shallahamer, *Forecasting Oracle Performance*. Berkeley: Apress, 2007.

[26] M. Ghazanfari, M. Jafari, and S. Rouhani, "A tool to evaluate the business intelligence of enterprise systems," *Scientia Iranica*, no. 0, pp. –, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S102630981100215X

[27] A. Nylund, "Tracing the BI Family Tree," *Knowledge Management*, 1999.

[28] H. J. Watson and B. H. Wixom, "The Current State of Business Intelligence," *Computer*, vol. 40, no. 9, pp. 96–99, 2007.

[29] S.-T. Li, L.-Y. Shue, and S.-F. Lee, "Business intelligence approach to supporting strategy-making of ISP service management," *Expert Syst. Appl.*, vol. 35, no. 3, pp. 739–754, 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1383655.1383758

[30] M. Tereso and J. Bernardino, "Open source business intelligence tools for SMEs," in *Information Systems and Technologies (CISTI), 2011 6th Iberian Conference on*, 2011, pp. 1–4.

[31] C. Thomsen and T. Pedersen, "A Survey of Open Source Tools for Business Intelligence," in *Data Warehousing and Knowledge Discovery*, ser. Lecture Notes in Computer Science, A. Tjoa and J. Trujillo, Eds. Springer Berlin / Heidelberg, 2005, vol. 3589, pp. 74–84. [Online]. Available: http://dx.doi.org/10.1007/11546849_8

[32] Pentaho, "Welcome to the Pentaho Community." [Online]. Available: http://community.pentaho.com/

[33] L. Ramos, "Performance Analysis of a Database Caching System In a Grid Environment," MSc Thesis (in portuguese), FEUP, 2007.

[34] K. Dias, M. Ramacher, U. Shaft, V. Venkataramani, and G. Wood, "Automatic performance diagnosis and tuning in Oracle," in *Proceedings of the 2005 CIDR Conf*, 2005. [Online]. Available: http://crmondemand.oracle.com/technetwork/database/focus-areas/manageability/cidr-addm-134903.pdf

[35] J. Beresniewicz, "Average active sessions: the magic metric," Oracle USA, Tech. Rep.

[36] Hoopoes, "Oracle Database Tuning Statistics," 2007. [Online]. Available: http://www.hoopoes.com/cs/oracle_tune.shtml

Intelligent Systems based in Hospital Database Malfunction Scenarios

Paulo Silva*, Cesar Quintas†, Pedro Gonçalves*, Gabriel Pontes*,
Manuel Santos‡, António Abelha§ and José Machado§
*Universidade do Minho, Department of Informatics, Braga, Portugal
†Centro Hospitalar do Porto, Porto, Portugal
‡Universidade do Minho, ALGORITMI, Guimarães, Portugal
mfs@dsi.uminho.pt
§Universidade do Minho, CCTC, Braga, Portugal
abelha,jmac@di.uminho.pt

***Abstract*—Databases are indispensable for everyday tasks in many organizations, particularly in healthcare units. Databases, allows to archive, among other relevant operations, important, private and confidential information about patients clinical status. Therefore, they must be available, reliable and at high performance level twenty-four hours a day, seven days per week. In many healthcare units, fault tolerant systems are online and ensure the availability, reliability and disaster recovery of data. However, these mechanisms do not allow to take preventive actions in order to avoid fault occurrence. In this context, it is of utmost importance the necessity of developing a fault prevention system. This system can predict database malfunction in advance and provides early decision taken to solve problems. With this paper we inted to monitor the database performance and adapt a forecasting model used in medicine (MEWS) to the database context. Based on mathematical tools it was created a scale that assesses the severity of abnormal situations. In this way, it is possible to define the scenarios where database symptoms must trigger alerts and assistance request.**

I. Introduction

A database is composed by a set of hardware and software components and is coordinated by the database management system (DBMS). This system is also responsible for managing the database users' information requests [1][2]. In healthcare units it is crucial that database and DMBS ensure: data confidentiality, data integrity and data availability [2][3][4].

This paper focuses on the issue of availability of data. In healthcare units databases store very important information about the patients' clinical status, administrative information and other relevant information for the healthcare services. As so this data should be available twenty-four hours a day, seven days per week [5]. However, ensuring the continuing availability of databases is not easy [6]. Faults often occur in the database. These faults can be of various types depending on the system where database are implemented. The most common type of faults in large databases under high utilization is related to resources limitation or high load. These faults occur when the physical resources are not enough to keep the database running. It is very important that the database must continue to operate despite the faults. Healthcare units have already implemented some fault tolerant systems, which intend to ensure the availability of data when a fault occurs. However, these systems only work after a fault occurs, not allowing to take preventive actions [7][8][9].

For that reason, the development of a fault forecasting and prevention system which allows taking early actions to solve future problems is crucial. Forecasting models have been used in critical areas such as medicine. The objective of this research is the development of a model for forecasting and prevention of database faults by adapting an existing forecasting model in medicine (Modified Early Warning Score - MEWS) [10].

This paper will present the information systems, the MEWS model, the methodology followed, the results obtained and finally the conclusions.

II. Healthcare Information Systems

The healthcare information systems allowed that, healthcare units, implement new methodologies for problem solving. These systems are responsible for the collection, storage, processing and data management of all stakeholders and all services. Thus, healthcare information systems enable a fast and efficient access to stored information always assuring data security [11].

Since the 1990s, hospital information systems in Portugal was implemented the SONHO (Hospital Information Integrated System). This system had as its objective merging clinical and administrative information. The evolution of this system is the genesis of the electronic medical record (EMR). An electronic medical record is a set of electronic documents that contain all information about patient health and his clinical risk profile [12] [13]. Besides the system already mentioned, in the healthcare units there are many other distributed and heterogeneous information systems. Therefore it is very difficult to ensure communication between the different systems. It became thus necessary to develop an effort to allow interoperability among all systems [13].

Therefore, a dynamic framework (AIDA- Agency for the Integration, Diffusion and Archive) was developed. This framework is composed of pro-active agents who are responsible for promoting communication. It made the management of all data most relevant to the process of interoperability

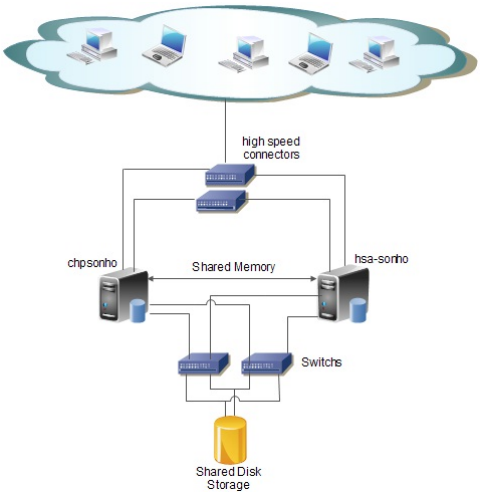


Fig. 1. SONHO RAC Architecture

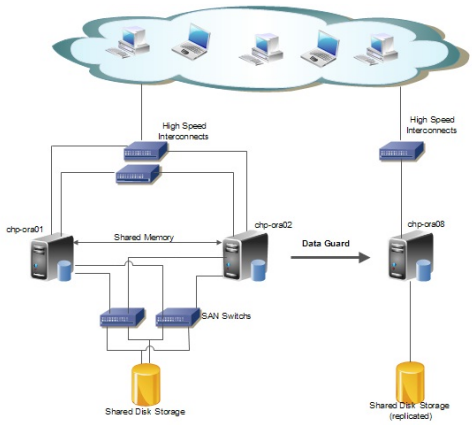


Fig. 2. AIDA Complete Architecture

between different systems. Thus, AIDA shares information and knowledge among every information system, namely: the administrative information system (AIS); the medical support information system (MIS); the nursing support information system (NIS); the EMR information system; the department information systems (DIS) of all the departments or of the healthcare unit [13][14].

The information is archived in very large databases which must be always available, because their information is vital for solving the patients' problems and for hospital management [5]. To achieve this objective fault tolerance mechanisms are used. These mechanisms can be based on the redundancy of data, components or both. They also can be used for load balancing or recovery [2][5].

The main databases (AIDA and SONHO) of Centro Hospitalar do Porto are based on an Oracle Real Application Clusters (RAC) System. This mechanism is provided by Oracle for improving the availability and scalability of databases. An example of these architectures can be see in Figure 1. These architectures are composed by a shared database which can be accessed through the server/computer that contains a database instance and an ASM (Automatic Storage Management) instance. So, it is possible access to the database across multiple servers. [2][8][15].

In AIDA database, there is also a data guard solution (Figure 2). A data guard solution consists in one or more standby databases (replicas of the original database) which should be in different places. Thus, when the master database is unavailable the replica can be used without the need to interrupt operation of the system. It is essential that the master and the standby databases are synchronized and the access is read-only during recovering [5].

AIDA and SONHO are very large databases. AIDA includes around 5451 tables, a large amount of sessions at same time and several agents responsible for ensuring interoperability. SONHO has around 2255 tables and a large number of sessions and processes that run permanently. Therefore, the most often faults in these databases are related to resource limitation. Then the focus of this paper is to implement a forecasting model thact predict faults associated with resource overhead.

III. MEWS

In medicine there is already used a model, the Modified Early Warning Score(MEWS), for the prediction, in advance, of serious health problems. This model assumes that a serious problem of health is often preceded by physiological deterioration. It uses a decision table, like the Table I, to evaluate the clinical status of the patient according the monitoring patients' vital signs values [10][16][17].

The Early Warning Score was been introduced in 1997 and the vital signs measured were: Temperature, Respiratory rate, Systolic BP, Heart rate and Neurological. However, some years later several studies have concluded that it was needed more parameters for the success of this model. In 1999 a new model appeared with the use of two additional variants: Urine output, Saturation of Hemoglobin with Oxygen SPO2 [16][17].

Normally, if any of the parameters have the score equal to two, the patient must be in observation. In the case of the sum of scores being equal to four, or there is an increase of two values in consecutive monitoring, the patient requires urgent medical attention. In a more extreme situation, if a patient has a score bigger than four, he is at risk of life [10][18].

The main advantage of this model is that it is possible identify the patients at risk and give them priority to access to the limited resources of intensive care units [16].

IV. Methodology

A. Monitoring database performance

The first step to build a predictive model of faults is to try to identify the source of faults. Therefore, it is necessary to monitor the database performance [19].

Monitoring is not an easy process, and its complexity increases when it is necessary to perform the monitoring of various components and systems with complex architectures. However, the Oracle systems provide several tools to help

TABLE I
MEWS SCORES

MEWS SCORE	**3**	**2**	**1**	**0**	**1**	**2**	**3**
Temperature (C)		< 35.0	35.1-36.0	36.1-38.0	38.1-38.5	> 38.6	
Heart rate (min1)		< 40	41-50	51-100	101-110	111-130	> 131
Systolic BP (mmHg)	< 70	71-80	81 - 100	101 - 199		> 200	
Respiratory rate (min1)		< 8		8-14	15-20	21-29	> 30
SPO2	< 85	85-89	90-93	> 94			
Urine output (ml/kg/h)	Nil	< 0.5					
Neurological		New confusion/agitation		Alert	Reacting to voice	Reacting to pain	Unresponsive

in this like performance views. These views contain useful information for monitoring. They show the current state of the database, so it is impossible to realize editing operations, it is only possible to consult information [20][21].

The most important views related to this paper are the views which contain information about database component statistics such as v$sysstat or v$sesstat [20][20].

It is also necessary to use others tools more related with physical resources. There are a lot of operative system commands that can help in the monitoring process but the most used are **Top**, **Sar**, **Ps**, **Vmstat**. Using these tools together we can do a more efficient database monitoring [22].

B. Business Intelligence(BI) tool

In this paper, the use of BI tools is useful because database monitoring provides a large quantity of data which are difficult to interpret and manipulate and also because the BI tools allow you to perform useful for dashboards show real-time forecasts.

The term Business Intelligence (BI) was introduced by Howard Dresner of the Gartner Group, to describe a set of concepts, methods, processes which utilize new technologies to improve the business, particularly, on decision making. BI tools it is also important for cost-cutting in data, time savings for user requests and more and better information [23][24].

After reviewing some studies about open source BI tools and realize experiments involving some of these tools, it was observed that Pentaho Community version is the most appropriate tool to use in this work [25][26].

The Community version of Pentaho provides several tools that are available for different computational platforms. These tools allow the creation of reports, dashboards and charts, application of data mining and integration techniques and data modeling [25][27]. There is a main application, bi-server, where it is possible to perform reporting and analysis. In this application, can be added a others plugs-in such as CDE that allows the development of dashboards. In this paper, the tools most frequently used were [27]:

- **CDE** - Allows the user to develop dashboards, in order to show more clearly the data collected/treated. It was developed and maintained by Webdetails. The front-end are based in HTML, and dashboards can be populated with data coming from a variety of sources, such: sql querys, xml files, mondrian cubes, spoon transformations [27].
- **Pentaho Data Integration (Spoon)** - Delivers a powerful Extraction, Transformation and Loading (ETL) capabilities. This tool is very useful to integrate information from different sources and also to perform some mathematical operations on data [27].

C. Explain of the statistics

There are several statistics that can be used to characterize the behavior of the database. Since the objective is to prevent fault in terms the resource limitation have been selected statistics related to the load from the database. These are [5][20]:

- **DB time** - This statistic give information about database response time. The response time is the period between an initial user requests until the return of the results. It is a good indicator of the workload of the system. Typically, this time increases with the number of simultaneous users or applications, but also may increase due to large transactions, or other system problems [5][28].
- **Number of transactions** - Transactions is units of work, i.e., the database have more or less work according with transactions which are realize. In Oracle databases, the number of transactions can be obtained by summing the values of statistics “user commits” and “user rollbacks” [5][6][22].
- **Number of executions** - One transaction can be result in a high set of operations in database depending of query. Therefore, it is important to collect information about the number of operations [22].
- **Calls ratio** - Calls can be of two types: user calls or recursive calls. The user call occurs when a user request can be resolved through a single SQL query. A recursive call occurs when a user request need more than one SQL query. Ideally this ratio should be as low as possible, since the high number of recursive calls can indicate problems with the design of tables. This ratio can be calculate by the equation [21][29]:

$$RC = \frac{(recursivecalls)}{(recursivecalls + usercalls)} \quad (1)$$

- **Number of current logons** - Logons not directly represents the number of users but the number of sessions, since a user may have multiple sessions. The collection

of this statistic is important because each session is associated with a piece of memory, so many simultaneous sessions can cause problems [20].

- **Processor utilization** - The processor is one of the most important statistic. Low values of processor utilization may indicate problems at the level of I/O. If the values are too high can compromise the functioning of the database [20].
- **Memory utilization** - The memory is a key component to the speed of the database systems, since, depending on whether the data are or are not in memory, the response time is influenced [6].
- **Size of redo file** - The redo files are used to store information about changes in the database. An increase in the size of these files, it indicates a higher number of operations and therefore a higher database load [21].
- **Buffer cache ratio** - This ratio show the percentage of data that are in memory cache, rather than disk. Normally, the BC is very high if BC decrease, this may indicate problems. BC can be calculate:

$$BC = \frac{1 - (physicalreads)}{(consistentegets + blockgets)} \qquad (2)$$

- **Amount of I/O requests** - The I/O operations spend a lot of time, and an excess of this kind of operations can indicate problems in memory. Due problems of the memory the access to disc is more frequent and more time is spent in I/O operations [20].
- **Amount of redo space requests** - Indicates the lack of space to write in the buffer, this can lead to delays because it is necessary to write some data to disk to release memory. This can happen due to a poorly sized buffer, or excess entries generated simultaneously [29].
- **Volume of network traffic** - The several database components are connected through network. If a volume of network is increasing greatly, the database can be slow and compromise users' requests [20].

D. Process

First, a java application was developed to collect data related with statistics described above from the two major databases (AIDA, SONHO) of Centro Hopitalar do Porto. For a week the monitoring program collected the values of mentioned statistics. With these values it is possible to create charts, in Pentaho, that demonstrate normal behavior of the database. For each statistic, we calculated the upper (using the 75th percentile) and lower (using the 25th percentile) limits. Thus, it is possible to detect abnormal situations as those statistics on the measured value is not within limits.

A new program was developed for catching the abnormal situations. Depending on the value of the deviation and the frequency of times that happen abnormal situations are assigned granted scores like as is done in MEWS.

According to these scores, alerts will be sent to database administrator and it is possible visible in dashboard. This program is upgradeable as new limits calculated at the end of the day based on new measurements.

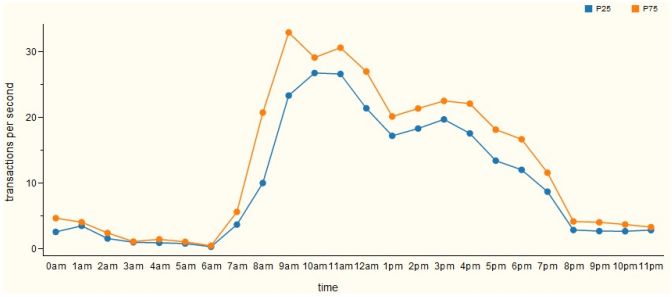


Fig. 3. Transactions per second in SONHO

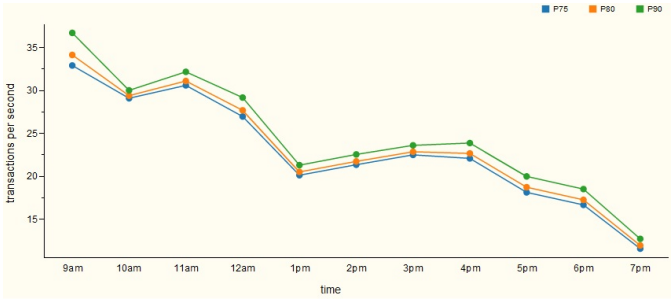


Fig. 4. Percentiles 75, 80 and 90

V. Results

The program is based in learning rules. Dynamically it creates a new model from normal database behavior. Figure 3 shows an example of a normal database (SONHO) behavior analyzing the statistic number of transactions per second. It is possible to see that the database has more transactions activity in the morning. Moreover, it is expected that 50% of the collect data are present between the two limits defined by the percentiles 25 and 75.

In order to create evaluation scores to identify the abnormal situations, it was calculated the percentiles 75, 80 and 90. The data above the percentile 75 are considered a low gravity situation, above 80 a grave situation and above 90 a critical situation. So, it is possible construct the decision table like in the Table II. The limits of the transactions per second can be seen in Figure 4.

For this test only are consider the period between 9am and 7pm because in this range there is the highest number of transactions per second.

Finally, the Figure 5 presents abnormal situations that occur along one day. According the previously scores, there is a low gravity situation at 9am and three critical situations at 11am, 12am and 1pm. In spite of these situations, the database continues in operation.

VI. Conclusions

Despite fault forecasting in healthcare databases is a complex theme, it was proposed a model to represent the normal

TABLE II
GRAVITY SCORES

SCORE	**0**	**1**	**2**	**3**
Value	$< p75$	$> p75 < p80$	$> p80 < p90$	$> p90$
Gravity	Normal	Low Gravity	Grave	Critical

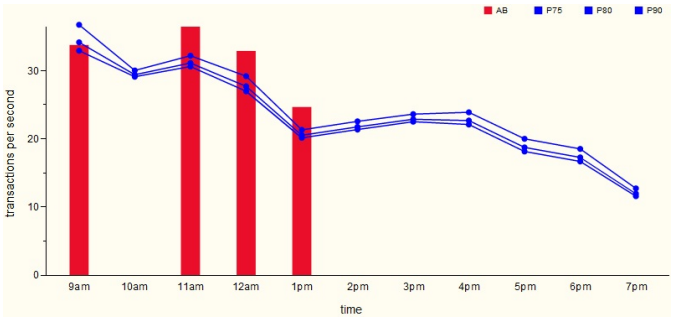


Fig. 5. Abnormal situations: Transactions per second in SONHO

workload for databases AIDA and SONHO, based some chosen statistics. It was also possible to identify peak usage of databases and periods of the day where the possibility of faults occurrence is bigger. With some mathematical tools it was possible to compute the limits for severity scores to characterize the abnormal situations. A decision table was created for this purpose. Thus, MEWS was adapted to the database context based in an upgradeable and learning forecasting fault system. The Pentaho Community tools were useful for manipulating data and for the development of the attractive dashboard for representing and interpretating results.

VII. ACKNOWLEDGMENT

This work is funded by National Funds through the FCT - Portuguese Foundation for Science and Technology within project PEst-OE/EEI/UI0752/2011.

REFERENCES

[1] T. Connolly and C. Begg, *Database Systems A Pratical Approach to Design, Implementation, and Management*, 4th ed. Harlow: Addison-Wesley, 2005.

[2] S. Drake, W. Hu, D. McInnis, M. Sköld, A. Srivastava, L. Thalmann, M. Tikkanen, O. y. Torbjø rnsen, and A. Wolski, "Architecture of Highly Available Databases," in *Service Availability*, ser. Lecture Notes in Computer Science, M. Malek, M. Reitenspieß, and J. Kaiser, Eds. Springer Berlin / Heidelberg, 2005, vol. 3335, pp. 1–16. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-30225-4_1

[3] A. Rodrigues, *Oracle 10g e 9i: Fundamentos Para Profissionais*. Lisboa: FCA, 2005.

[4] E. Bertino and R. Sandhu, "Database security - concepts, approaches, and challenges," *Dependable and Secure Computing, IEEE Transactions on*, vol. 2, no. 1, pp. 2–19, 2005.

[5] R. Godinho, "Availability, Reliability and Scalability in Database Architecture," MSc Thesis, Universidade do Minho, 2011.

[6] R. Schumacher, *Oracle Performance Troubleshooting With Dictionary Internals SQL & Tuning Scripts*. Kittrell: Rampant TechPress, 2003.

[7] O. Yi, "Highly Available Database Systems," in *Integrated information systems and databases seminar*, 2006.

[8] M. Cyran, P. Lane, and J. Polk, *Oracle Database Concepts, 10g Release 2 (10.2)*. Oracle, 2005. [Online]. Available: http://docs.oracle.com/cd/B19306_01/server.102/b14220.pdf

[9] W. Hodak, S. Kumar, and A. Ray, "Oracle Database 11g High Availability," Oracle, Tech. Rep., 2007.

[10] C. P. Subbe, M. Kruger, P. Rutherford, and L. Gemmel, "Validation of a modified Early Warning Score in medical admissions," *QJM*, vol. 94, no. 10, pp. 521–526, 2001. [Online]. Available: http://qjmed.oxfordjournals.org/content/94/10/521.abstract

[11] E. Ammenwerth, J. Brender, P. Nykänen, H.-u. Prokosch, M. Rigby, and J. Talmon, "Visions and strategies to improve evaluation of health information systems Reflections and lessons based on the HIS-EVAL workshop in Innsbruck," *International Journal of Medical Informatics*, 2004.

[12] S. Lameirão, "Hospital management and use of information systems: application to CHVR-PR," MSc Thesis (in portuguese), Universidade de Trás-os-Montes e Alto Douro, 2007.

[13] J. Machado, V. Alves, A. Abelha, and J. Neves, "Ambient intelligence via multiagent systems in the medical arena," *Engineering Intelligent Systems for Electrical Engineering and Communications*, vol. 15, no. 3, pp. 151–157, 2007.

[14] J. Machado, A. Abelha, P. Novais, J. Neves, and J. a. Neves, "Quality of service in healthcare units," *Int. J. Computer Aided Engineering and Technology*, vol. 2, no. 4, pp. 436–439, 2010.

[15] M. Bauer and R. Strohm, *Oracle Real Application Clusters Administration and Deployment Guide, 11g Release 1 (11.1)*. Oracle, 2009. [Online]. Available: http://docs.oracle.com/cd/B28359_01/rac.111/b28254.pdf

[16] A. Albino and V. Jacinto, "Implementation of early warning score - EWS," Centro Hospitalar do Barlavento Algarvio, EPE, Portimão (in portuguese), Tech. Rep., 2009.

[17] J. Gardner-Thorpe, N. Love, J. Wrightson, S. Walsh, and N. Keeling, "The value of Modified Early Warning Score (MEWS) in surgical in-patients: a prospective observational study." *Annals of the Royal College of Surgeons of England*, vol. 88, no. 6, pp. 571–5, Oct. 2006.

[18] G. Devaney and W. Lead, "Guideline for the use of the modified early warning score (MEWS)," Outer North East London Community Services, Tech. Rep., 2011.

[19] S. Nair, "The Art of Database Monitoring," *Information Systems Control Journal*, no. Ccm, pp. 1–4, 2008.

[20] I. Chan, *Oracle Database 2 Day + Performance Tuning Guide, 10g Release 2 (10.2)*. Oracle, 2010.

[21] K. Rich, *Oracle Database Reference, 10g Release 2 (10.2)*. Oracle, 2009.

[22] C. Shallahamer, *Forecasting Oracle Performance*. Berkeley: Apress, 2007.

[23] M. Ghazanfari, M. Jafari, and S. Rouhani, "A tool to evaluate the business intelligence of enterprise systems," *Scientia Iranica*, no. 0, pp. –, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S102630981100215X

[24] A. Nylund, "Tracing the BI Family Tree," *Knowledge Management*, 1999.

[25] M. Tereso and J. Bernardino, "Open source business intelligence tools for SMEs," in *Information Systems and Technologies (CISTI), 2011 6th Iberian Conference on*, 2011, pp. 1–4.

[26] C. Thomsen and T. Pedersen, "A Survey of Open Source Tools for Business Intelligence," in *Data Warehousing and Knowledge Discovery*, ser. Lecture Notes in Computer Science, A. Tjoa and J. Trujillo, Eds. Springer Berlin / Heidelberg, 2005, vol. 3589, pp. 74–84. [Online]. Available: http://dx.doi.org/10.1007/11546849_8

[27] Pentaho, "Welcome to the Pentaho Community." [Online]. Available: http://community.pentaho.com/

[28] J. Beresniewicz, "Average active sessions: the magic metric," Oracle USA, Tech. Rep.

[29] Hoopoes, "Oracle Database Tuning Statistics," 2007. [Online]. Available: http://www.hoopoes.com/cs/oracle_tune.shtml

Hospital database workload and fault forecasting

***Abstract*—With the growing importance of hospital information systems, databases became indispensable tools for day-to-day tasks in healthcare units. They store important and confidential information about patient's clinical status and about the other hospital services. Thus, they must be permanently available, reliable and at high performance. In many healthcare units, fault tolerant systems are used. They ensure the availability, reliability and disaster recovery of data. However, these mechanisms do not allow the prediction or prevention of faults. In this context, it emerges the necessity of developing a fault forecasting system. The objectives of this paper are monitoring database performance to verify the normal workload for the main database of Centro Hospitalar do Porto and adapt a forecasting model used in medicine into the database context. Based on percentiles it was created a scale to represent the severity of situations. It was observe that the critical workload period is the period between 10:00 am and 12:00 am. Moreover, abnormal situations were detected and it was possible to send alerts and to request assistance.**

I. Introduction

The evolution of technology contributed to the exponential growth of data which needs to be stored for the proper functioning in organizations. Databases are powerful tools where this large amount of data can be stored and managed in a simple way. The database management system (DBMS) coordinates a set of hardware and software components, furthermore, DBMS is also responsible for managing the database users' requests [1].

Today, databases are considered essential for everyday organizations tasks. In healthcare units, due to hospital databases store very important information about the patients' clinical status, administrative information and other relevant information for the healthcare services, they are essential. Therefore, it is crucial to ensure the confidentiality, integrity and availability of data [1][2][3].

However, database availability is a complex feature. There are several problems that influence database availability, such as: the database can be inaccessible due to network problems or a virus; the database can be too slow and therefore not satisfy the user's requests [4].

The unavailability of the healthcare units databases is often related resource limitation faults. These faults occur when the physical resources are not enough to keep the database running. It is essential that the database must continue to operate despite the faults. For this reason, fault tolerant systems are already used. They are responsible to ensure the availability of data even a fault occurs. However, these systems do not allow the early detection of faults [5][6][7].

Therefore, the development of a fault forecasting system which allows taking early actions to solve problems is crucial. Forecasting models have been used in critical areas such as medicine. The objectives of this research is characterized the workload of the one database of Centro Hospitalar do Porto and a development of a model for forecasting and prevention of database faults by adapting an existing forecasting model in medicine (Modified Early Warning Score) [8].

The remaining paper is organized in the following sections: in the second section, we will address the issue of hospital information systems, showing the platform where our study was made ; in the third, we will present the forecast model (MEWS); At the fourth, will be described the methodology. In the fifth part, the results will be presented and discussed. At last, in seventh section the conclusions are presented.

II. Hospital Information Systems

HIS can be defined as a subsystem hospital with a socio-technological development, which covers all hospital information processing[9]. Its main purpose is to contribute to the quality and efficiency of healthcare. This objective is primarily oriented to the patient after being directed to health professionals as well as the functions of management and administration [9][10]. The HIS also assumes much importance in relation to costs since the sector of communication technologies in healthcare is increasingly important [10]. There are four basic functional processes. This process list begins with the patient's admission and ends in discharge or transfer to another institution. The other categories will serve to support the healthcare with the primary objective of improving quality. The four functional categories are characterized as follows: care; clinical process management; work organization and resource planning; and hospital management [11][12].

EHR can be assumed as a HIS for excellence and has replaced the traditional manual recording in Paper Clinical Process (PCP). EHR may include all hospital areas with a need for registration information. This information can be clinical, administrative and financial [13][14].

A. AIDA

AIDA means Agency for Integration, Diffusion and Archive of Medical Information. It is a platform that consists of a Multi-Agent System (MAS) and it can be considered like the main HIS where it has been working. The AIDA main goal is to overcome difficulties in achieving uniformity of clinical systems, as well as medical and administrative complexity of different hospital information sources [14]. AIDA was created by a group of researchers from the University of Minho and is currently installed in many Portuguese hospitals. It is an electronic platform that holds intelligence electronic employees (agents). This platform promotes a proactive behavior in its main functions: communication between

hospital heterogeneous systems; storage and management all hospital information; response to requests in time; sending and receiving information from hospital sources like laboratories (medical reports, images, prescriptions). Thus, AIDA enables interoperability between hospital subsystems, assuming a main role where it is installed [15][16]. AIDA has an easy access for your users, allowing the management of clinical information anywhere in the hospital. In addition, the platform enables the sending of messages via phone or e-mail. The same way, AIDA establishes connection with all others systems of patients information: EHR; Administrative Information System (AIS) used by administrative people; Medical Information System (MIS) used in medical record; and Nursing Information System (NIS) used by nurses [17].

B. AIDA-PCE

The AIDA-PCE is an EHR and was implemented in the Centro Hospitalar do Porto. It is working as a subsystem of the main HIS. The AIDA-PCE follows a problem-oriented organization suggested by Lawrence Weed in the 60s. This information organization is known as the Problem Oriented Medical Record (POMR) and it assumes that registration is a production of clinical scientific document. In this kind of organization, the clinical information (annotations, therapeutics and treatments, diagnostics, diaries) should be recorded for specific problem, creating a list of issues organized in a tree structure, where each new problem derives from the main branch [13][18][19]. These problems must be classified as active or inactive, in which active problems are those where the disease is still active or even when medical intervention is required immediately. On the other hand, inactive problems require no urgent action. In these EHR problems assets are monitored and recorded daily using a SOAP (Subjective, Objective, Assessment and Planning) framework. Thus, each record contains the patient's Symptoms, a doctor's Observation, an Analysis of diagnosis and a treatment Plan that the patient is subject to [13][19]. The AIDA-PCE has many common features with PCP but it has a response, which is fast, reliable and safe. The structure of this EHR allows seamless integration with existing HIS by promoting the ubiquity of records between different specialties and services. The ubiquity of the AIDA-PCE allows access to mechanisms for monitoring alarm systems and decision support. The electronic record allows generation of documents and customized reports for specific purposes. It becomes easier to configure interfaces for registration and more. The information contained herein is standardized and uniform [13][15][20]. In the hospitals where AIDA and AIDA-PCE was installed, it was made a substantial investment to ensure the availability, reliability and scalability of the system.

C. AIDA and AIDA-PCE faults

As it was mentioned above, AIDA platform and AIDA-PCE are running at real behavior with real cases, because of this reason some problems have emerged. The problems are not responsibility of the AIDA and AIDA-PCE but these ones interfere directly with AIDA and AIDA-PCE operation and, consequently, the quality of the medical record and patient's treatment. The main problems are due to communication faults in the network and are related to peak hours using it. Other situations are related to database faults, such as: bad management of datacenter; many users accessed at the same time; breaks on energy. Some of these crashes must to be avoided, but actually errors will always happen [1].

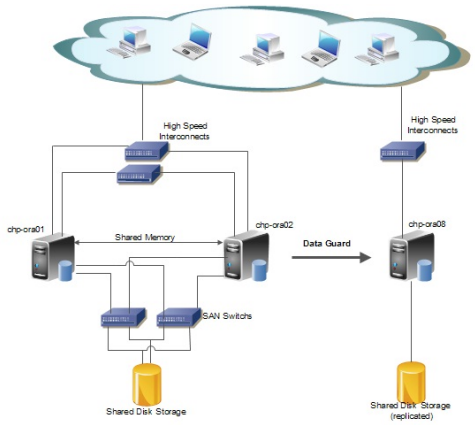


Fig. 1. AIDA and AIDA-PCE Architecture

D. AIDA and AIDA-PCE fault tolerance mechanism

Fault tolerance mechanisms are used to ensure availability of data but also to load balancing and recovery [1][2].

AIDA and AIDA-PCE architecture, present in Figure 1, is composed by an Oracle Real Application Clusters System (RAC) and a dataguard solution. RAC is used for improving the availability and scalability. This architecture is composed by a shared repository of data which can be accessed through the server/computer that contains a database instance. If a server is down, it is possible access to the database easily by another server [1][6].

A data guard solution consists in one or more standby databases (replicas of the main database). When the main database is unavailable the replica can be used without the need to interrupt operation of the system. It is crucial that the main and the standby databases are synchronized and the access is read-only during recovering [2].

III. MEWS

The concept of Early Warning Score was been introduced in 1997 and the vital signs measured were: Temperature, Respiratory rate, Systolic BP, Heart rate and Neurological. However, in 1999 a new model appeared with the use of two additional variants: urine output, saturation of hemoglobin with oxygen SPO2 [21][22]. This model is the Modified Early Warning Score (MEWS). MEWS assumes that a serious problem of health is often preceded by physiological deterioration.

This model consists in a strict and continuous monitoring of the patient's vital signs. Then, using the decision table, see Table I, the scores are calculated to determine the level of risk

TABLE I
MEWS SCORES

MEWS SCORE	**3**	**2**	**1**	**0**	**1**	**2**	**3**
Temperature (C)		< 35.0	35.1-36.0	36.1-38.0	38.1-38.5	> 38.6	
Heart rate (min -1)		< 40	41-50	51-100	101-110	111-130	> 131
Systolic BP (mmHg)	< 70	71-80	81 - 100	101 - 199		> 200	
Respiratory rate (min -1)		< 8		8-14	15-20	21-29	> 30
SPO2	< 85	85-89	90-93	> 94			
Urine output (ml/kg/h)	Nil	< 0.5					
Neurological		New confusion/agitation		Alert	Reacting to voice	Reacting to pain	Unresponsive

of each patient, trying to understand when a serious problem will be occur [8][21] [22].

Normally, if any of the parameters have a score equal to two, the patient must be in observation. In the case of the sum of scores being equal to four, or there being an increase of two values the patient requires urgent medical attention. In a more extreme situation, if a patient has a score higher than four, his life is at risk. The use of this model allows identify the patients at risk and give them priority, improves the monitoring physiological parameters of the patients and thus support the medical decision [8][23].

IV. METHODOLOGY

A. Monitoring database performance

Monitoring is the first step in the process of identify the source and symptoms of the faults. This is a complex process, however, Oracle provide several tools to help in this process. An example of these tools are the **performance views** which contain useful information for monitoring [24][25].

The most important views related to this paper are the ones which contain information about database component statistics. For global monitoring of the database must be used system-level views (v$sysstat, v$sys_time_model, v$system_event). For more detailed information about the sessions should be used to session level views (v$sesstat,v$session_event, v$sess_time_model [4][25]. It was also used an operating system command (*sar*) to gather relating to the memory and processor utilization [26].

There are others tools for database monitoring. However, the performance views were chosen for two reasons: they make the integration with the business intelligence tool used to do the analysis and presentation of data (Pentaho) easier; and it is easier to define the zone that is to be monitored, thereby reducing the cost associated with monitoring. A Java application was developed to monitoring AIDA-PCE database.

According to the objective of preventing faults related to the resource limitation have been selected the following statistics [2][24][27]:

DB time - This statistic gives information related to database response time. The response time is the period between an initial user request and the return of the results. In Oracle systems, this time is a sum of total time (including CPU time, IO time, Wait time). Therefore it is a good indicator of the workload of the system. Typically, this time increases with the number of simultaneous users or applications, but it also may increase due to large transactions [2][28].

Number of transactions - Database had more or less work according to the transactions which are performed. In Oracle databases, the number of transactions can be obtained by adding up the values of statistics “user commits” and “user rollbacks” due each transaction always ends with a “commit” command and any undo operation as a “rollback” command [2][4][26].

Number of Operations - One transaction may trigger a large set of operations (sum of user calls and recursive calls) depending the query. This means that there may be a large number of transactions and a low number of operations or otherwise. In Oracle databases this information can be obtained by collecting, the “execute count” statistic [26].

Number of sessions - The collection of this statistic is important because each session is associated with a piece of memory, so many simultaneous sessions can cause problems. In Oracle systems, the number of sessions can be obtained by statistic “logons current” [24].

Processor utilization - The processor is one of the most important components, so it is necessary to constantly monitor its utilization by the user processes. Low values of processor utilization may indicate problems at the level of I/O. If the values are too high, it can compromise the functioning of the database. The percentage of utilization can be obtained thought a command of operating system such as **sar -u**[24].

Memory utilization - The memory is a key component to the speed of the database systems. The speed of access to data depends on the place where they are: memory or disk. If the data are in memory then access to them is faster. This statistic is also accessible through the operating system commands such as **sar -r** [4].

Size of redo file - The redo files are used to store information about changes made to the database. These are very important for the recovery of faults. An increase in the size of these files, it indicates a higher number of operations and therefore a higher database load. In Oracle systems, the size (kb) of redo file can be obtained by statistic “redo size” [25].

Amount of I/O requests - The I/O operations are very time consuming often are associated with writing or reading data from memory to disk. An excess of this kind of operations

can indicate problems in memory [24].

Amount of redo space requests - Indicates the lack of space to write in the redo buffer, this can lead to delays because it is necessary to write some data to disk to release memory. This can happen due to a poorly sized buffer, or excess entries generated simultaneously [29].

Volume of network traffic - Network is very important for database performance because several database components are connected through network, and all the user requests come from the network. If a volume there is a problem in network, the database can be slow and compromise users' requests [24].

Recursive Calls ratio - Calls to database can be of two types: user calls or recursive calls. When a user request can be resolved through a single SQL query, this is a call. A recursive call occurs when a user request need one query SQL that needs another SQL query. Ideally this ratio should be as low as possible. The recursive calls ratio is the fraction between the recursive calls and total calls (user calls + recursive calls). The high value can indicate problems with the design of tables or an excessive amount of triggers running at the same time. This ratio can be calculated by the equation [25][29].

Buffer cache ratio (BC) - This ratio shows the percentage of data that is in memory cache, rather than in the disk. Normally, the BC is very high so it is necessary to pay attention if BC decreases, this may indicate lack of memory problems. BC can be calculated by the fraction between the number of disk accesses (physical reads) and the number of memory accesses (consistente gets + block gets).

The monitoring program collected values during a month. Using a Business Intelligence tool it is possible analyze the data collected and create graphs that demonstrate the normal behavior of the database. For each statistic, were calculated the upper (using the 75th percentile) and lower (using the 25th percentile) limits.

B. Business Intelligence (BI) tool

BI tools are useful because database monitoring provides a large quantity of data which is difficult to analyze. Thus, BI tools allow the management of the data and present the conclusions of a more clearly way.

The BI tool chosen for this research was Pentaho Community Version because it has all the desired features. The Pentaho Community version provides tools that allow the creation of reports, dashboards/charts, application of data mining, integration techniques and data modeling [30][31]. There is a main application, the bi-server, where it is possible to perform reporting and analysis. In this application, a plug-in which allows the development of dashboards can be added. However, other applications can be easily integrated with the main server in order to add new features. The applications used in this paper are [31]:

- **CDE** - Allows the user to develop the updateable dashboards, in order to show more clearly the data collected/treated. It was developed and maintained by Webdetails. The front-end is based on HTML, and dashboards can be populated with data coming from a variety of sources, such: SQL querys, xml files, mondrian cubes, spoon transformations [31].
- **Pentaho Data Integration (Spoon)** - Delivers powerful Extraction, Transformation and Loading (ETL) capabilities. This tool is very useful to integrate information from different sources and also to perform some mathematical operations on data. The division of a multi-step transformation reduces the complexity of the SQL query used to obtain the desired information [31].

Show 10 entries Search:

Name	Average (node1)	Average (node2)	Total (node1+node2)	Comparison (nodo1-nodo2)
DB time per second	3.015	2.987	6.002	
Network Traffic Volume (bytes/sec)	380657.63	373348.54	754006.17	
Number of operations per second	414.634	404.997	819.631	
Number of I/O requests per second	376.897	365.629	742.526	
Number of sessions	340.554	333.892	674.446	
redo log space requests per second	0.005	0.005	0.01	
Number of redo size (kb/s)	106909.058	49782.597	156691.655	
Transactions per second	146.136	144.424	290.56	

Showing 1 to 8 of 8 entries

Fig. 2. AIDA Workload

After represented the normal behavior of the database, a new program was developed for detecting the abnormal situations. Depending on the value of the deviation, for each statistic, is assigned scores to abnormal situations such as is done in MEWS. Two situations can happen (see Table II): if the score is greater than 0 and less than 3, a visual warning will be issued on the dashboard; if the score is greater than 3, warnings will be sent via email to the database administrator, allowing he to take speedy action to prevent the occurrence of a fault in the database.

This program is upgradeable, as new limits calculated at the end of the day based on new measurements, because it is possible that exists an increase of load and the database remains operational.

V. RESULTS

As a first step will be examined workload of the database. In Figure 2, is represented a table that contain the mean values of each statistical measure for each node. It is possible observe that node 1 (chp-ora01) has more workload than node2 (chp-ora02). Only in the "redo log space requests per second", the mean of node1 is not higher to the node 2.

Overall, the AIDA database has an average 674 sessions and about 290 transactions per second, which shows that is a database with a high workload. TThe high amount of

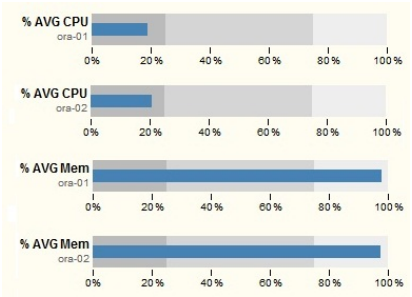


Fig. 3. AIDA Workload

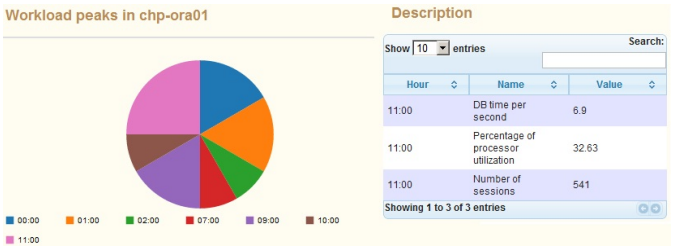


Fig. 4. Workload peaks chp-ora01 node

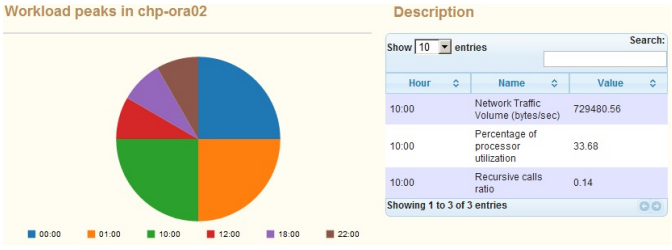


Fig. 5. Workload peaks chp-ora02 node

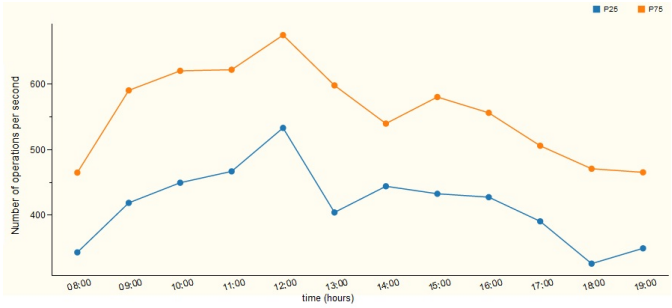


Fig. 6. Limits of number of operations per second in chp-ora01

network traffic indicates that this is an important metric for evaluating the database performance. Concerning the use of two main components, memory and processor, through the Figure 3 is possible to observe that the average is practically the same in both nodes. Memory presents high values unlike the processor. This may be due to take into account only the processor utilization by user processes. To identify the most critical points of the day, graphs and tables of Figures 4, 5 were constructed. In Figure 4, it is possible identify that the period between 11:00 and 12:00 is the most critical period in node1. In this period, there is three important peaks: DB time, percentage of processor usage and number of sessions.

Figure 5, presents the workload peaks for node two (chp-ora02). In this node, the distribution of peaks is more diverse. However, in the period between 10:00 and 11:00, there is three important peaks: Network traffic, percentage de processor usage and recursive call ratio. In general one can consider the period from 10:00 to 12:00, as the period most favorable for the occurrence of faults. For each statistic of each node limits were calculated by using percentiles. The Figure 6, shows the normal behavior of the variable "number operations per second" throughout the day. For this example only the period between 08:00 and 19:00 is considered because in this range there is the highest number operations per second. It is possible to see that the database has more activity in the morning, due to the higher number of users connected. It is expected that 50% of the collected data are present between the two limits defined by the 25th and 75th percentiles. In order to identify the abnormal situations scores were assigned to percentiles. The data above the percentile 75 is considered a low severity situation, above 80 a grave situation and above 90 a critical situation (see Table II). Figure 7 presents abnormal situations that occur along one day. In the period between 12:00 and 14:00 there is two abnormal situations. The first is a severe situation and the other is a critical situation, according to Table II. However, these situations do not cause database fault. For this reason it is necessary to update the limits. Limits are update in the end of day taking account of all measured values which do not cause fault.

VI. Conclusions

A model was proposed to characterize and represent the normal workload for AIDA database. It was observed that the node1 of the database is more used than node2. Moreover, workload peaks were identified and it was possible to observe that the critical period is the period between 10:00 am and 12:00 am.

With percentiles, it was possible to determine the limits that characterize the abnormal situations. A decision table, with percentiles and scores was created. Therefore, the methodology of MEWS was adapted to database reality through an upgradeable and learning forecasting fault system. The Pentaho Community allowed performing the data analysis and development of dashboards which makes it easy the interpretation of results these results.

Due to the heterogeneity of the database workload it was found that the limits will be updated every day.

Acknowledgment

This work is financed with the support of the Portuguese Foundation for Science and Technology (FCT), with the grant SFRH/BD/70549/2010 and within project PEst-OE/EEI/UI0752/2011.

TABLE II
SEVERITY SCORES

SCORE	**0**	**1**	**2**	**3**
Value	$< p75$	$> p75 < p80$	$> p80 < p90$	$> p90$
Severity	Normal	Low severity	Severe	Critical

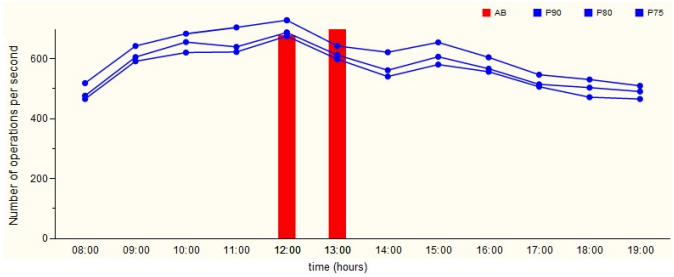


Fig. 7. Limits and abnormal situations in chp-ora01

REFERENCES

[1] S. Drake, W. Hu, D. McInnis, M. Skold, A. Srivastava, L. Thalmann, M. Tikkanen, O. y. Torbjornsen, and A. Wolski, "Architecture of Highly Available Databases," in *Service Availability*, ser. Lecture Notes in Computer Science, M. Malek, M. Reitenspie, and J. Kaiser, Eds. Springer Berlin / Heidelberg, 2005, vol. 3335, pp. 1–16. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-30225-4_1

[2] R. Godinho, "Availability, Reliability and Scalability in Database Architecture," MSc Thesis, Universidade do Minho, 2011.

[3] A. Rodrigues, *Oracle 10g e 9i:Essentials For Professionals (in portuguese)*. Lisboa: FCA, 2005.

[4] R. Schumacher, *Oracle Performance Troubleshooting With Dictionary Internals SQL & Tuning Scripts*. Kittrell: Rampant TechPress, 2003.

[5] O. Yi, "Highly Available Database Systems," in *Integrated information systems and databases seminar*, 2006.

[6] M. Cyran, P. Lane, and J. Polk, *Oracle Database Concepts, 10g Release 2 (10.2)*. Oracle, 2005. [Online]. Available: http://docs.oracle.com/cd/B19306_01/server.102/b14220.pdf

[7] W. Hodak, S. Kumar, and A. Ray, "Oracle Database 11g High Availability," Oracle, Tech. Rep., 2007.

[8] C. P. Subbe, M. Kruger, P. Rutherford, and L. Gemmel, "Validation of a modified Early Warning Score in medical admissions," *QJM*, vol. 94, no. 10, pp. 521–526, 2001. [Online]. Available: http://qjmed.oxfordjournals.org/content/94/10/521.abstract

[9] R. Haux, A. Winter, E. Ammenwerth, and B. Brigl, *Strategic Information Management in Hospitals: An Introduction to Hospital Information Systems*. Springer-Verlag, 2004.

[10] J. Machado, A. Abelha, P. Novais, J. Neves, and J. a. Neves, "Quality of service in healthcare units," *Int. J. Computer Aided Engineering and Technology*, vol. 2, no. 4, pp. 436–439, 2010.

[11] J. Duarte, C. F. Portela, A. Abelha, J. Machado, and M. F. Santos, "Electronic health record in dermatology service," in *Communications in Computer and Information Science, 221 CCIS (PART 3)*, 2011.

[12] H. AmmenwerthE., BuchauerA., "Arequirementsindexfor information processing in hospitals," *Methods of Information in Medicine*, vol. 41, pp. 282–288, 2002.

[13] M. Miranda, G. Pontes, P. Gonçalves, H. Peixoto, M. Santos, A. Abelha, and J. Machado, "Modelling intelligent behaviours in multi-agent based hl7 services," in *Studies in Computational Intelligence*, Springer, Ed., vol. 317, 2010.

[14] E. Coiera, *Guide to Health Informatics (2nd ed.)*, 2nd ed. London: Hodder Arnold, 2003.

[15] J. Duarte, J. Neves, A. Cabral, M. Gomes, V. Marques, M. F. Santos, A. Abelha, and J. Machado, "Towards intelligent drug electronic prescription," in *European Simulation and Modelling Conference*, Guimarães, Portugal, 2011.

[16] F. Portela, M. Vilas-Boas, M. F. Santos, A. Abelha, J. Machado, A. Cabral, and I. Aragao, "Electronic health records in the emergency room," in *ACIS-ICIS 2010*, pp. 195–200.

[17] V. Slee, D. Slee, and J. Schmidt, *The Endangered Medical Record: Ensuring Its Integrity in the Age of Informatics*. Saint Paul, Minnesota: Tringa Press, 2000.

[18] M. Miranda, J. Duarte, A. Abelha, J. Machado, and J. Neves, "Interoperability and healthcare," in *European Simulation and Modelling Conference*, Leicester, UK, 2009.

[19] K. Hyrinen, K. Saranto, and P. Nyknen, "Definition, structure, content, use and impacts of electronic health records: A review of the research literature," *International Journal of Medical Informatics*, vol. 77, pp. 291–304, 2008.

[20] C. Bossen, "Evaluation of a computerized problem-oriented medical record in a hospital department: Does it support daily clinical practice?" *International Journal of Medical Informatics*, vol. 77, pp. 592–600, 2007.

[21] A. Albino and V. Jacinto, "Implementation of early warning score - EWS," Centro Hospitalar do Barlavento Algarvio, EPE, Portimão (in portuguese), Tech. Rep., 2009.

[22] J. Gardner-Thorpe, N. Love, J. Wrightson, S. Walsh, and N. Keeling, "The value of Modified Early Warning Score (MEWS) in surgical in-patients: a prospective observational study." *Annals of the Royal College of Surgeons of England*, vol. 88, no. 6, pp. 571–5, Oct. 2006.

[23] G. Devaney and W. Lead, "Guideline for the use of the modified early warning score (MEWS)," Outer North East London Community Services, Tech. Rep., 2011.

[24] I. Chan, *Oracle Database Performance Tuning Guide, 10g Release 2 (10.2)*. Oracle, 2008.

[25] K. Rich, *Oracle Database Reference, 10g Release 2 (10.2)*. Oracle, 2009.

[26] C. Shallahamer, *Forecasting Oracle Performance*. Berkeley: Apress, 2007.

[27] L. Ramos, "Performance Analysis of a Database Caching System In a Grid Environment," MSc Thesis (in portuguese), FEUP, 2007.

[28] K. Dias, M. Ramacher, U. Shaft, V. Venkataramani, and G. Wood, "Automatic performance diagnosis and tuning in Oracle," in *Proceedings of the 2005 CIDR Conf*, 2005. [Online]. Available: http://crmondemand.oracle.com/technetwork/database/focus-areas/manageability/cidr-addm-134903.pdf

[29] Hoopoes, "Oracle Database Tuning Statistics," 2007. [Online]. Available: http://www.hoopoes.com/cs/oracle_tune.shtml

[30] M. Tereso and J. Bernardino, "Open source business intelligence tools for SMEs," in *Information Systems and Technologies (CISTI), 2011 6th Iberian Conference on*, 2011, pp. 1–4.

[31] Pentaho, "Welcome to the Pentaho Community." [Online]. Available: http://community.pentaho.com/