

MONITORAMENTO DE INFRAESTRUTURA DE TI: MÉTRICAS E ESTUDOS EXPERIMENTAIS

Felipe da Motta Varga ¹

Prof. Dr. Kleinner Silva Farias de Oliveira ²

Resumo: Manter uma infraestrutura de Tecnologia da Informação (TI) funcionando de forma estável é crucial para o sucesso do negócio de empresas que oferecem produtos e serviços via internet. Além disso, o monitoramento do funcionamento de TI e a tomada de decisões são duas atividades essenciais para a manutenção do negócio das empresas. Atualmente o monitoramento de TI não se restringe à simples coleta de informações básicas sobre o estado do funcionamento da infraestrutura de TI. Existem diversas ferramentas capazes de coletar métricas de disponibilidade, de performance, status de máquina virtual Java, por exemplo. Uma prática encontrada no mercado corporativo é a utilização de ferramentas especializadas de monitoramento de TI para que se possa extrair o melhor do que elas têm a oferecer. Neste contexto, surgem como uma tendência corporativa o uso de técnicas de *software analytics*, a qual utiliza dados coerentes de forma centralizada, para possibilitar aos públicos interessados uma consulta mais fácil com o objetivo de explorar e analisar de forma perspicaz e objetiva. Com este trabalho foi criada então uma base de dados (*Data Warehouse*) onde se armazenam de forma centralizada os valores de métricas e thresholds, possibilitando além da consulta unificada, uma visão histórica de alterações.

Palavras-chave: Software Analytics. Data Warehouse. Business Intelligence.

Abstract: Maintain an Information Technology (IT) infrastructure working in a stable manner is crucial to the success of companies business that offer products or services via the Internet. At this way, monitoring the behaviour of IT and make decisions based on information collected by this monitoring is crucial to keep the business of companies working in a stable manner . Currently IT monitoring is not restricted to the simple collect of basic information about the operation state of the technological environment or infrastructure. There are several tools that can collect metrics of availability, performance, status of Java virtual machine, for example. A practice found in the corporate market is the use of specialists tools for monitoring IT so you can get the best out of what they have to offer. In this context, appears as a corporative tendency to use techniques of software analytics, which uses centrally and consistent data, to enable stakeholders easy reference for the purpose of exploring and analyzing perceptively and objectively. With this work has been created a data base (*Data Warehouse*) where the metrics and thresholds values are stored in a centralized way, making beyond the unified consult, a hitorical vision of changes.

Keywords: Software Analytics. Data Warehouse. Business Intelligence.

¹ Graduando do curso de Ciência da Computação – UNISINOS – Universidade do Vale do Rio dos Sinos. E-mail: fmvarga@gmail.com

² Orientador do presente trabalho – UNISINOS – Universidade do Vale do Rio dos Sinos. E-mail: kleinner@gmail.com

1 INTRODUÇÃO

O monitoramento de infraestrutura de TI desempenha um papel fundamental na manutenção dos serviços das empresas na internet. Um exemplo deste serviço seria os serviços disponibilizados por bancos, incluindo acompanhamento de movimentações financeiras, transferência de valores entre contas, entre outros. Para que este monitoramento seja possível, as empresas fazem uso de uma grande diversidade de ferramentas para conseguir cobrir os diferentes aspectos e tecnologias utilizadas para manter os serviços sempre disponíveis para os seus clientes. Estas ferramentas são capazes de, por exemplo, capturar ou receber diferentes métricas dos ativos de TI. As equipes de monitoramento das empresas, por sua vez, usam os valores gerados por tais métricas para potencializar o entendimento sobre os possíveis riscos associados à disponibilidade dos serviços oferecidos pelas empresas. É importante destacar que, se os serviços não são disponibilizados para os clientes, então as empresas deixam de gerar receita. Logo, o monitoramento de infraestrutura de TI é crítico para o sucesso dos negócios das empresas.

As equipes de monitoramento utilizando *thresholds* para auxiliar na identificação de possíveis cenários críticos — aqueles em a indisponibilidade de algum serviço (ou de todos) se torna latente devido a eventos e incidentes indesejados. Se uma métrica atinge valores acima do estabelecido por um *threshold*, então tem-se identificado um provavelmente cenário que poderá comprometer algum serviço oferecido. Atualmente, dada a quantidade cada vez maior de recursos computações que devem ser monitorados em uma infraestrutura de TI, as equipes de monitoramento têm utilizado o recurso de “alertas” para facilitar a identificação de tais cenários. Esses alertas são gerados em uma *console* única, também conhecida como Gerente dos Gerentes¹ (Dragich, 2013).

O problema é que existe um ponto central de monitoramento dos dados gerados pelas métricas, porém as métricas e os *thresholds* estão especificados diretamente nas ferramentas. Consequentemente, é desafiante estabelecer um controle aceitável das inclusões, alterações e remoções dos valores associados às métricas e aos *thresholds*. Um outro agravante é que, na prática, membros da equipe

¹ Do inglês *Manager of Managers*

de monitoramento acabam alterando os valores dos *thresholds* para reduzir o número de alertas gerados. Porém, tais ferramentas não armazenam o histórico das métricas, muito menos o histórico das alterações dos valores dos *thresholds*, impossibilitando, desse modo, um controle do histórico das alterações dos valores dos *thresholds* e das métricas.

Este trabalho, portanto, tenta mitigar tais problemas ao apresentar o projeto e a implementação de uma aplicação para centralizar a coleta das métricas e manter o histórico das alterações dos *thresholds*. Para isso, utilizou-se a técnica de ETL (*Extract/Transform/Load*) para extrair, transformar e carregar os dados localizados em diferentes máquinas para uma base de dados centralizada (um *Data Warehouse*). Combinando tais elementos, espera-se não apenas organizar e qualificar as informações existentes nos diversos sistemas de monitoramento de infraestrutura de TI, mas também utilizando-se de técnicas de *Software Analytics* agregar valor às áreas da empresa interessadas nestes resultados.

1.1 Formulação do Problema

Atualmente o monitoramento de TI não se restringe à coleta de informações sobre o estado do funcionamento da infraestrutura de TI. Pelo contrário, há diversas ferramentas capazes de coletar métricas de disponibilidade, de performance, status de máquina virtual Java, por exemplo. Uma prática encontrada no mercado corporativo é a utilização de ferramentas especialistas de monitoramento de TI para que se possa extrair o melhor do que cada ferramenta oferece.

Na prática, para o operador de monitoramento é fornecida uma única ferramenta com o objetivo de centralizar os alertas provenientes de várias outras ferramentas, como citado anteriormente. Tal prática centraliza os alertas em si, mas não fornece informações sobre como e por que tais alertas estão sendo gerados. Uma consequência disto é que dificulta a consulta e a identificação dos fatores causadores dos incidentes na infraestrutura de TI. Desse modo, o grande desafio atualmente é que as ferramentas utilizadas não fornecem informações estratégicas para os tomadores de decisões. Isto é devido a algumas razões: (1) as ferramentas coletam e armazenam dados separadamente; (2) os dados coletados não são

confrontados visando inferir conhecimentos implícitos nos mesmos; e (3) os dados coletados são voláteis e os *thresholds* utilizados, bem como suas alterações, raramente podem ser justificadas. É importante destacar a importância dos *thresholds*, pois, uma vez que eles são atingidos, alguma ação deve ser tomada visando algum objetivo que usualmente não é claro para o operador de monitoramento.

1.2 Limitações dos trabalhos relacionados

As grandes limitações dos trabalhos relacionados dizem respeito ao escopo dos trabalhos e a aplicabilidade dos mesmos. Ao estudar o estado da arte em *Software Analytics*, percebe-se que os trabalhos são focados na aplicação de técnicas em projetos de software com o objetivo de qualificar as informações pertinentes ao desenvolvimento de ferramentas. Neste sentido, pretende-se aplicar tais técnicas em um ambiente real de desenvolvimento visando criar uma base de dados através da coleta de dados distribuídos. Os trabalhos relacionados procuram em diversas bases sobre um escopo restrito de software as informações e no caso deste trabalho existe a situação onde serão coletadas as informações em diversas ferramentas utilizadas com um objetivo em comum: o monitoramento da infraestrutura de TI.

1.3 Objetivos

O objetivo deste trabalho consiste na consolidação e no armazenamento das informações geradas a partir do monitoramento de infraestrutura de TI em uma base de dados única, com uma interface que facilite a consulta e permita o cruzamento dos dados. Mais especificamente, neste trabalho foi desenvolvida uma base de dados com os dados das métricas e “*thresholds*” de ferramentas de monitoramento de TI, bem como foi implementada uma ferramenta manipular esta base de dados gerada.

1.4 Metodologia do Estudo

O trabalho foi realizado em etapas, onde cada etapa teve um conjunto de macro-atividades agrupadas pela dependência de execução de suas subatividades. Estas etapas foram executadas seguindo um fluxo pré-definido visando garantir o cumprimento de um cronograma proposto. É importante destacar que todos os métodos de pesquisa que serão aplicados seguirão orientações presentes no estado-da-arte na área de Engenharia de Software Experimental (Menzies, 2010) (Wohlin, 2000). As etapas do estudo são apresentadas a seguir e a sua ordem de apresentação implica na ordem de execução:

Etapas 1: esta etapa focou na execução de atividades visando fomentar o entendimento dos aspectos teóricos do trabalho, do estado da arte, e na caracterização do problema a ser atacado no TC1 e no TC2. Em particular, nesta etapa foram executadas as seguintes atividades: (1) estudo dos aspectos teóricos do trabalho; (2) estudo do estado-da-arte; (3) caracterização da problemática do estudo; e (4) formulação das questões de pesquisa. Para isto, artigos científicos, teses, dissertações e livros sobre *software analytics*, *data Warehouse* e ETL foram lidos.

Etapas 2: baseado no entendimento adquirido na Etapa 1, esta etapa se concentrou no desenvolvimento ou uso de uma ferramenta de extração, transformação e carga de dados, bem como centralização de informações de métricas e “*thresholds*”, assim como na definição de um método de avaliação da ferramenta visando quantificar o seu custo-benefício. Em particular, nesta etapa foram executadas as seguintes atividades: (5) análise e projeto da ferramenta, incluindo estudo de viabilidade do uso de alguma ferramenta já existente no estado da arte para solucionar os problemas encontrados no monitoramento de infraestrutura de TI; e (6) definição do método de avaliação que permitiria avaliar o custo-benefício do uso da ferramenta, bem como viabilizou a realização de um estudo de caso em um ambiente real.

Etapas 3: esta etapa se concentrou na elaboração da monografia do TC1. Ela se caracterizou como uma etapa que ocorreu em paralelo às Etapas 1 e 2, pois a

finalização das atividades daquelas etapas implica em fornecer os requisitos necessários para execução parcial (ou total) das atividades desta etapa, as quais são descritas a seguir: (7) redação da monografia do TC 1; e (8) revisão e entrega da monografia.

Etapa 4: após a conclusão do TC1, foi inicializada a (9) implementação da ferramenta proposta e avaliada no TC1 com as funcionalidades propostas e englobando pelo menos duas ferramentas de monitoramento de infraestrutura de TI para que sejam efetuadas as extrações, transformações e armazenamento dos dados de métricas e *thresholds* de tais ferramentas. Para o estudo de caso proposto neste trabalho, foram escolhidas as ferramentas CA eHealth e Oracle Enterprise Manager em virtude de ambas incluírem a maior parte de cobertura de monitoramento no ambiente alvo. Tendo sido concluída a etapa 9 será então (10) confeccionado um artigo conforme requisitos do TC2 para que posteriormente seja realizada (11) apresentação à banca avaliadora.

1.5 Contribuições

As contribuições deste trabalho são geradas a partir da execução das etapas desta pesquisa descrita anteriormente que permitirá resolver, ou ao menos minimizar, os problemas apontados na Seção 1.2. Além disso, estas contribuições avançam o estado da arte por mitigar as limitações dos trabalhos relacionados apontados na Seção 1.3. Tais contribuições são descritas a seguir: (1) uma análise do estado da arte sobre o uso de métricas como ferramenta de monitoramento de infraestrutura de TI; (2) análise e projeto de uma ferramenta para coleta, tratamento e visualização centralizada e consistente dos dados de monitoramento; (3) conhecimento empírico e lições apreendidas sobre as dificuldades encontradas (Seção 1.1) para a centralização e geração de informações detalhadas e consistentes. É importante ressaltar que essas contribuições estão alinhadas com as tendências corporativas no que se refere ao uso de técnicas de *Software Analytics* (Buse & Zimmermann, 2012) (Davenport, 2010)(Poster, 1990) para gerar informações coerentes e centralizadas sobre o estado atual de infraestrutura de TI.

2 REFERENCIAL TEÓRICO

Após apresentar a formulação do problema abordado, indicar as limitações dos trabalhos relacionados, descrever os objetivos e a metodologia de estudo, bem como as contribuições esperadas com este trabalho, esta seção foca em descrever os principais conceitos que são necessários para o bom entendimento sobre monitoramento de infraestrutura de TI. Para isso, o conteúdo apresentado nas próximas seções abordará sobre monitoramento de TI (Seção 2.1) e sobre *Software Analytics* (Seção 2.2).

2.1 Monitoramento de TI

Dada a alta competitividade entre as empresas que oferecem produtos ou serviços via internet, manter uma infraestrutura de Tecnologia da Informação (TI) funcionando de uma forma estável é um fator determinante para o sucesso ou insucesso dos negócios. Uma consequência direta disso é que a manutenção do bom funcionamento da infraestrutura de TI adquiriu o status de missão crítica. Exemplos destas empresas seriam sites de comércio eletrônico e instituições financeiras. Se esta infraestrutura deixa de funcionar, ou não funciona da forma esperada, as empresas deixam de lucrar, visto que os seus produtos e serviços deixam de ser comercializados. Desse modo, monitorar o funcionamento de TI e tomar decisões baseadas nas informações coletas deste monitoramento é de fundamental importância para manter o negócio das empresas em funcionamento de uma maneira estável.

Sistemas de monitoramento de TI auditam constantemente vários ativos de TI com o objetivo de verificar o seu status (ativo ou inativo) e checar problemas de performance. Se problemas ocorrem, então alguém deve ser notificado para tomar medidas corretivas, bem como medidas preventivas para evitar problemas futuros. Exemplos de ativos de TI que são críticos para as organizações são: servidores, serviços de rede, aplicações, serviços web, dispositivos, sites da internet, links de comunicação, roteadores, e dentre outros.

A finalidade da auditoria desses ativos de TI é que a empresa possa ser reativa, proativa ou ainda tenha um papel preditivo quanto aos possíveis problemas que possam vir a ocorrer nos seus ativos. Atuar dessa forma é crucial para manter os serviços oferecidos pelas empresas aos seus clientes, parceiros, colaboradores e fornecedores. Para isso, se faz necessário ter um amplo monitoramento com o objetivo de identificar o comportamento e desempenho desses ativos de TI, bem como fornecer insumos sobre os pontos mais críticos de funcionamento, aqueles que, uma vez não respeitados, comprometerá algum serviço oferecido pela empresa. Os insumos gerados são importantes, por exemplo, para identificar pontos de melhorias e de direcionamento de investimentos, e tomada de decisão quanto aos rumos tecnológicos da empresa. Com um sistema de monitoramento eficaz e proativo, evita-se ações que necessitem de atitudes não planejadas para correção de incidentes, permitindo a detecção de problemas na sua fase inicial. Isso evidencia a necessidade da detecção de tais problemas em sua fase inicial ou antes da sua ocorrência, prevenindo o aumento do impacto.

O monitoramento de TI também visa indiretamente melhorar a experiência do usuário, visto que interrupções do funcionamento dos serviços causam complicações e descontentamento para o cliente e normalmente acabam levando a uma perda de confiança. Detectando problemas e sua causa raiz, permite aos administradores de sistemas a tomada de medidas corretivas (paliativas ou definitivas) para restabelecimento das plenas funcionalidades. Além disso, a redução de custos e maximização da utilização dos recursos também é possível, visto que permite a destinação de equipes especializadas às tarefas nobres e produtivas voltadas aos problemas de negócio reais, enquanto os sistemas de monitoramento se encarregam de operacionalizar os testes e validações quanto ao bom funcionamento do ambiente tecnológico. Outro benefício trazido por um amplo monitoramento de TI, a flexibilidade e o gerenciamento de complexidades são os benefícios trazidos com um amplo monitoramento de TI.

Para produzir todos esses benefícios, os ambientes de TI atuais exigem soluções multi-plataforma, formando, conseqüentemente, um ambiente heterogêneo de operação. A utilização de diversas soluções de nicho ou scripts complexos para monitorar estes ambientes resulta em rigidez causada por fatores secundários. As

soluções de monitoramento modernas oferecem suporte para diversas plataformas, e proporcionam flexibilidade que é um quesito indispensável para os negócios, além de serem interligáveis possibilitando a unificação dos alertas de forma única, mesmo se utilizando de diversas soluções.

Sob a ótica de gestão, é importante controlar o desempenho da TI e planejar sua capacidade: Hoje em dia, as decisões sobre os negócios estão diretamente vinculadas às de TI. As melhores soluções de monitoramento oferecem painéis de visualização que demonstram o desempenho da TI em tempo real. Também fornecem dados necessários para acompanhar o histórico dos sistemas e serviços de TI, para justificar custos e para planejar proativamente os investimentos.

Dada a grande variedade e heterogenia de sistemas e plataformas faladas anteriormente, em empresas de grande porte, acaba-se por optar pela utilização de ferramentas especialistas de monitoramento, onde cada ferramenta estará encarregada de obter uma maior quantidade de dados e evidências através de métricas específicas. Tais métricas são coletadas em frequência predefinida (*polling*) e de acordo com os limites (*thresholds*) definidos geram seus alertas. Visto a existência de diversas ferramentas especialistas, é necessária uma console central de eventos que auxilie na visualização e correta correlação de alarmes. Esta função é de suma importância, pois um evento detectado em mais de uma ferramenta deve ser correlacionado pelo operador de monitoramento ao analisar a amplitude e impacto dos alertas recebidos.

Atualmente o monitoramento de TI não se restringe à simples coleta de informações básicas sobre o estado do funcionamento do ambiente tecnológico, ou infraestrutura de TI. Existem diversas ferramentas capazes de coletar métricas de disponibilidade, de performance, status de máquina virtual Java, status de banco de dados, estatísticas de acesso externo e de usuários, falhas de *hardware*, temperatura de *hardware*, umidade do local, por exemplo. Uma prática encontrada no mercado corporativo é a utilização de ferramentas especialistas de monitoramento de TI para que se possa extrair o melhor do que elas têm a oferecer.

Na prática, para o operador de monitoramento é fornecida uma console única, uma ferramenta com o objetivo de centralizar os alertas provenientes das outras. Tal

prática centraliza os alertas em si, mas não fornece dados de como (métricas) ou por que (*thresholds*) tais alertas estão sendo gerados, dificultando a consulta e o processo de correlação ou ainda decisório quanto às alterações a serem feitas ou de fatores causadores de incidentes de infraestrutura de TI. Desse modo, o grande desafio atualmente é que as ferramentas aplicadas não são eficientes ao ponto de fornecer informações estratégicas para os tomadores de decisões. Isto é devido a algumas razões: (1) as ferramentas coletam e armazenam dados separadamente; (2) os dados coletados não são confrontados visando inferir conhecimentos implícitos nos dados; e (3) os dados são voláteis e os coletar informações relevantes para os tomadores de decisões, os dados coletados são voláteis e os *thresholds* utilizados raramente podem ser justificáveis, ou mesmo se tem um histórico dos motivos das atribuições dos valores dos *thresholds* utilizados. É importante destacar a importância dos *thresholds* pois, uma vez que eles são atingidos, alguma ação de ser tomada visando algum objetivo que muitas vezes não é claro para o operador de monitoramento.

2.2 Software Analytics

Software Analytics é uma área com um crescimento expressivo nos últimos anos. Isto se deve à tendência de crescimento de diversas bases de dados não centralizadas e à necessidade de minerar dados presentes em tais bases e gerar informação consistente e completa seja para evitar maiores duplicações de dados, seja para tomadas de decisão baseadas em tais dados. Pode ser definida como a análise em dados de softwares para gestores e engenheiros de software, com o objetivo de capacitar indivíduos e equipes visando aumentar/compartilhar uma visão de seus dados e assim auxiliar na melhor tomada de decisões. Tais visões propiciam o aconselhamento em como direcionar investimentos em upgrades, atacar pontos de contenção e visualização de problemas de capacidade e disponibilidade sob o contexto de monitoramento de TI.

Grandes empresas têm adotado uma cultura “analítica” para auxiliar os seus processos de tomada de decisão. Dentre os exemplos, encontramos empresas como IBM, Google e Microsoft, que buscam através de *Software Analytics* responder perguntas críticas para os rumos a serem trilhados através de visões baseadas em fatos, processos e tecnologias (Zhang, 2012). Tais análises podem ser usadas para

melhorar a competitividade das empresas. Abordagens analíticas representam esforços para oferecer visões em tempo real, e essas ideias são muitas vezes apresentadas como relatórios multidimensionais quantitativos. No entanto, tais abordagens podem ser quantitativas e qualitativas na sua natureza. Enquanto a análise quantitativa tem como destaque as tendências em dados de alto nível, as análises qualitativas permitem a tomada de decisão em tempo real para tarefas que possuem nível inferior e de maior frequência. Sob a ótica de *Software Analytics*, são justamente as análises qualitativas que permitirão uma visão do todo, com dados coletados de locais distribuídos e relacionados entre si, proporcionando uma análise e decisão baseados em todo o contexto.

Consciência situacional é um termo da psicologia cognitiva que refere-se a um estado de espírito em que a pessoa está ciente de elementos do seu ambiente imediato, tem uma compreensão quanto ao seu significado, e pode antecipar (ou planejar mudar) esses elementos em um futuro próximo. O termo é utilizado na engenharia, por exemplo, para descrever como os controladores de tráfego aéreo trabalham e como eles controlam o tráfego aéreo e rotas, mas também é uma boa descrição de como os profissionais de TI devem manter consciência do que está acontecendo em seu ambiente de trabalho, de como eles devem gerenciar um fluxo constante de informações e reagir em conformidade aos acontecimentos.

Com o eminente crescimento e distribuição de dados, pessoas envolvidas com TI estão se voltando para uma visão analítica destes dados. Isto significa o uso de análise, dados e fundamentação sistemática para a tomada de decisões. *Software Analytics* é a aplicação de uma visão analítica em dados de software para engenheiros e gerentes com o objetivo de capacitar os desenvolvedores e compartilhar visões de tais dados para que sejam tomadas melhores decisões. Tais visões incluem, mas não estão limitadas a como melhorar projetos de software, mas também embasar investimentos em infraestrutura, avaliação sob a ótica da gestão de disponibilidade, capacidade, ativos, eventos e incidentes. Estas gestões estão presentes na metodologia ITIL e podem usar dados relevantes obtidos através do monitoramento de infraestrutura de TI para embasar suas opiniões e também as decisões referentes a cada área. Devido ao grande volume de dados, a atividade de encontrar e agrupar estas visões tipicamente envolve um alto grau de automação

combinado com envolvimento humano. Em diversos estudos da área de *Software Analytics* já se demonstrou seu poder e resultados expressivos em várias vertentes no sentido de reunir e apresentar informações que agreguem valor na tomada de ações.

Para a tomada de decisão baseado em dados apresentados, é importante que as informações sejam reais e indiquem as ações a serem efetuadas. Se o gestor está se encaminhando para um ponto crítico, então o mesmo deve receber informações que sejam importantes para resolver os problemas que levaram à situação crítica, ou ao menos possa reduzir o problema. Por exemplo, se um gestor se encaminha para um penhasco, não devemos distraí-lo com dados sobre as nuvens no céu ou as flores na beira da estrada. Ao invés disto, baseado em indicações inteligentes, devemos gritar no seu ouvido: “Ei, existe um penhasco logo adiante, vire imediatamente para a esquerda”. Esse exemplo ilustra que tão importante quanto avisar sobre o obstáculo, é também informar sempre que possível a melhor escolha a ser tomada no momento, baseado na análise sobre os dados. O que acontecerá após, é de responsabilidade do gestor, que pode escolher ignorar o aviso e tomar a decisão baseado puramente em seus instintos.

É importante ter em mente que se não forem entregues informações relevantes e precisas, existe pouca esperança quanto ao uso futuro das informações provenientes desta origem. Na prática, isto significa que as informações provenientes da análise dos dados precisam estar disponíveis em tempo real e com o menor tempo de atualização possível, pois as decisões precisam ser tomadas baseadas em dados reais e atualizados de acordo com a possível velocidade de atualização dos dados das bases coletadas. Em particular, *Software Analytics*, está focado em o que o software pode aprender sobre si mesmo e sobre os outros sistemas de software, utilizando dados em comum e relevantes.

Melhor do que procurar por modelos globais, o foco atual é na procura de métodos locais. Ao final do século 20, foram disponibilizadas uma nova geração inteira de algoritmos para mineração de dados. Juntamente com uma imensa quantidade de dados provenientes de programas de código aberto (Mens, 2002). Como consequência, mais pesquisadores aplicaram mais mineração ao maior montante de dados disponíveis. Um efeito secundário desta situação foi a

visualização de que um modelo único não conseguiria cobrir todos os projetos de forma efetiva e assertiva. Antigamente se esperava que um modelo cobrisse toda a gama de programas e que fosse possível compartilhar isto, mas se verificou que seria melhor a utilização de modelos específicos para cada situação e isto se comprovou pelo reuso de modelos que levaram a decisões inapropriadas e com alto custo de gestão. Recentemente na publicação do “Inductive Engineering Manifesto” (Menzies,2011) foram publicados alguns itens que caracterizam a diferença entre a mineração de dados acadêmica e a aplicação prática efetuada pelo mercado de TI.

De qualquer forma é importante também iniciar a aplicação de *Software Analytics* de uma forma exploratória, onde nem sempre os resultados iniciais serão claros e agregarão valor ao negócio, mas após algumas adaptações e evoluções será possível ter conclusões preliminares sobre a validação do resultado. Após alcançar alguns pontos favoráveis ao que se espera da ferramenta será possível evoluir o uso de forma a cada vez mais agregar visões sobre o apresentado e evoluindo a uma segunda fase, onde os objetivos se tornam mais claros, bem como a maneira de alcançá-los. Será, então, o momento onde o time envolvido na construção da ferramenta, na mineração de dados e integração deve aumentar pois algumas tarefas podem tomar grande tempo e as expectativas quanto a prazo de desenvolvimento se reduzem.

Exemplos de tarefas que podem ter o uso de *Software Analytics* são: identificação de problemas de desempenho, entendimento do uso de padrões de uma funcionalidade específica em um programa e selecionar casos relevantes em uma grande massa de dados com origens distintas. Informação detalhada transmite entendimento ou conhecimento significativo e útil para execução de uma tarefa alvo. Tipicamente isto não pode ser facilmente obtido por investigação direta sobre os dados brutos, sem o auxílio de *Software Analytics*. Informações passíveis de ação são aquelas sobre as quais os profissionais de TI podem chegar a soluções concretas (melhores do que as já existentes, se houverem) no sentido de realização da tarefa alvo.

Artefatos de software normalmente carregam semânticas específicas do domínio do software e assim sendo, não podem ser tratados simplesmente como

dados genéricos como texto e sequencias. Entender a semântica de tais artefatos de software é pré-requisito para a posterior análise dos dados.

Na prática, entender os dados tem três partes: interpretação, seleção e filtragem. Para conduzir a interpretação dos dados, os desenvolvedores precisam entender as definições básicas específicas do domínio em questão, suas terminologias, significados e conceitos. Para conduzir a seleção de dados, os desenvolvedores precisam entender as conexões existentes entre os dados e o problema a ser resolvido. Para conduzir a filtragem dos dados, é necessário que os desenvolvedores entendam os defeitos e limitações dos dados existentes para evitar inferências incorretas ou que destoem do desejado.

3 PROJETO DA FERRAMENTA

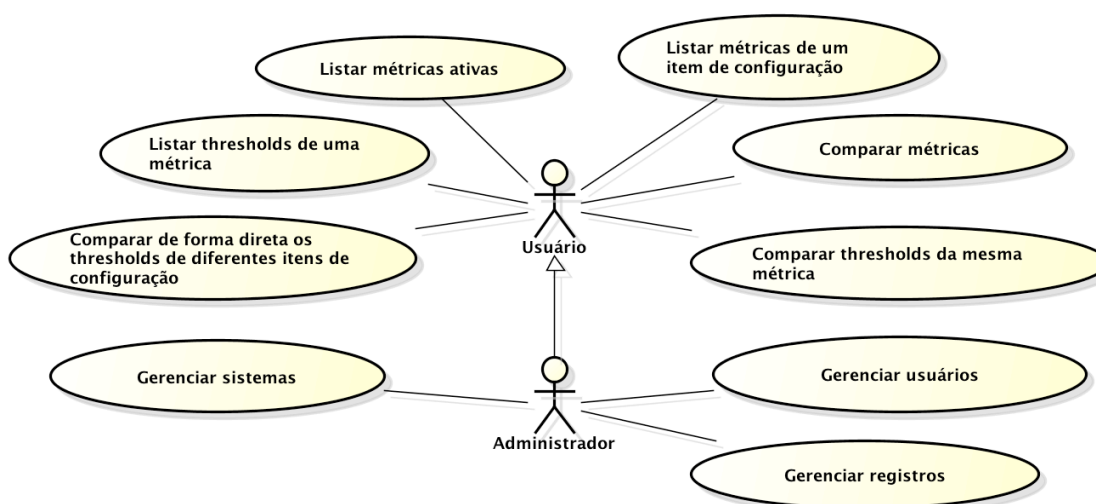
Esta Seção apresenta o projeto da ferramenta implementada neste trabalho, a MonaLisa. A ferramenta apresenta um conjunto de melhorias em relação aos trabalhos reportados na Seção de Trabalhos Relacionados (Seção 5), tais como: (1) a centralização do armazenamento dos dados gerados; (2) realização de ETL; (3) quantificação de métricas e de *thresholds*, bem como disponibiliza uma interface amigável para consulta; e, por fim, (4) faz uso de premissas de *Software Analytics* para reduzir os problemas de descentralização e análise de dados durante o monitoramento da infraestrutura de TI. As melhorias permitirão, por exemplo, encontrar de forma sistemática informações que auxiliem na gestão e tomada de decisão. É importante destacar que a ferramenta foi projetada a partir de cenários reais de uso em uma instituição financeira, na qual o autor tem trabalhado nos últimos anos.

O projeto da ferramenta é baseado nos artefatos propostos pela UML (*Unified Modeling Language*). A UML foi escolhida pois, segundo Miles e Hamilton (2008), ela é a linguagem padrão de modelagem de sistemas. A versão a ser utilizada na modelagem do presente trabalho será a versão 2.4.1, disponibilizada pela OMG em agosto de 2011 e publicada em abril de 2012 como as normas ISO/IEC 19505-1:2012 e ISO/IEC 19505-2:2012.

3.1 Requisitos

A Figura 1 apresenta o diagrama de casos de uso da ferramenta, o qual apresenta de forma abstrata as principais funcionalidades da ferramenta. É importante destacar que estas funcionalidades apresentadas foram derivadas de um estudo detalhado do estado da arte e do referencial teórico, bem como da experiência do autor com o monitoramento de infraestrutura de TI em instituições financeiras ao longo dos anos. Todos os requisitos foram identificados, analisados, e posteriormente testados através de estudos experimentais em uma instituição financeira de grande porte.

Figura 1 Diagrama de caso de uso do MonaLisa.



Fonte: Elaborada pelo autor do sistema.

Os requisitos funcionais e não-funcionais são descritos a seguir. Enquanto os requisitos funcionais definem como o sistema deve reagir às entradas específicas e como o sistema deve se comportar em determinadas situações, os requisitos não-funcionais definem as restrições sobre as funcionalidades oferecidas pelo sistema como, por exemplo, restrições de tempo, processo de desenvolvimento e padrões.

3.1.1 Requisitos funcionais

Os requisitos funcionais da ferramenta são apresentados sucintamente abaixo:

- **RF01 - Gerenciar usuários:** o administrador poderá adicionar, consultar, editar e excluir usuários do sistema;
- **RF02 - Gerenciar sistemas:** o administrador poderá adicionar, consultar, editar e excluir sistemas de monitoramento;
- **RF03 - Gerenciar registros:** o administrador poderá adicionar, consultar, editar e excluir registros de uma determinada métrica do sistema;
- **RF04 - Listar métricas de um item de configuração:** o usuário poderá listar todas as métricas de um determinado item de configuração;
- **RF05 - Listar *thresholds* de uma métrica:** o usuário poderá listar todos os registros de *thresholds* de uma determinada métrica, ordenando os dados por data de criação do registro ou valor;
- **RF06 - Listar métricas ativas:** o usuário poderá listar todas as métricas que estão ativos nos sistemas integrados;
- **RF07 - Listar *thresholds* ativos:** o usuário poderá listar todos os *thresholds* que estão ativos nos sistemas integrados;
- **RF08 - Comparar métricas:** o usuário poderá visualizar e comparar métricas existentes para diferentes itens de configuração, a fim de auxiliar na tomada de decisão e possíveis alterações;
- **RF09 - Comparar de forma direta os *thresholds* de diferentes itens de configuração:** o usuário poderá visualizar e comparar *thresholds* de uma mesma métrica para diferentes itens de configuração, a fim de auxiliar na tomada de decisão e possíveis alterações;
- **RF10 - Comparar *thresholds* da mesma métrica:** o usuário poderá visualizar e comparar *thresholds* (atual e anteriores) da mesma métrica existente para um item de configuração específico, para fins de identificação de alterações recentes e possível correlação com incidentes, problemas de capacidade, disponibilidade, e dentre outros.

3.1.2 Requisitos não-funcionais

Após apresentar os requisitos funcionais, os requisitos não-funcionais da ferramenta são descritos abaixo. É importante destacar que tais requisitos visam definir como a ferramenta tratará os seguintes atributos de qualidade, tais como: desempenho, usabilidade, confiabilidade, segurança, disponibilidade e manutenibilidade. Os requisitos não-funcionais são:

- **RNF01 - Escalabilidade e confiabilidade:** este sistema deve operar de forma a proporcionar a possibilidade do servidor de banco de dados e de aplicação serem separados e de existência de um ou mais servidores de aplicação para o mesmo banco de dados;
- **RNF02 - Interoperabilidade - utilizar servidor web:** o servidor MonaLisa será uma aplicação web para consulta ao DW;
- **RNF03 – Interoperabilidade - ETL:** o sistema possibilitará a conexão aos diversos tipos de fontes de dados, tais como bancos de dados (por exemplo MySQL, Oracle, SQL Server, etc), bem como arquivos textos para coletar e armazenar informações de métricas e *thresholds*;
- **RNF04 - Interoperabilidade - DW:** o serviço de DW permitirá sua consulta através de um sistema web que deve ser escolhido para realização de consultas, utilização e administração do sistema como um todo;
- **RNF05 - Portabilidade:** o sistema será disponibilizado em plataforma web e assim sendo, permitindo a utilização de consultas em dispositivos móveis, *tablets* ou *smartphones*;
- **RNF06 - Segurança - uso de certificados digitais:** será possível a opção pela utilização de certificados digitais para criptografar a comunicação entre os aplicativos servidor e cliente;
- **RNF07 - Segurança - uso de senha:** os usuários utilizarão senhas que lhes permitirão acessar de forma segura o sistema.
- **RNF08 - Segurança - integração:** o sistema deverá permitir a integração com sistemas de autenticação centralizada (LDAP, *Active Directory*, etc), permitindo assim uma administração centralizada de senhas;
- **RNF09 - Usabilidade - browser:** o sistema deve permitir a utilização de pelo menos dois browsers diferentes existentes no mercado;

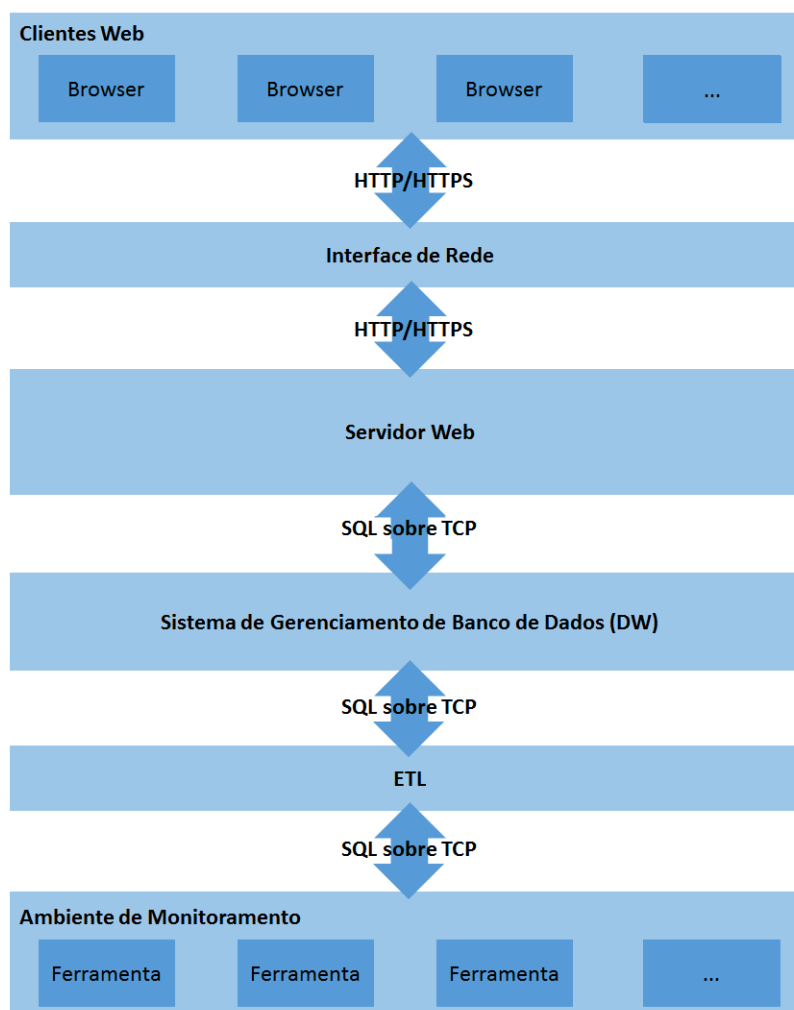
- **RNF10 - Usabilidade – exportação de dados:** a aplicação deverá possibilitar a exportação dos relatórios elaborados em formato eletrônico ou impresso.

3.2 Arquitetura

O MonaLisa foi construído com o conceito de sistema web. A aplicação será acessada através de um navegador que irá se conectar ao servidor da camada web e este irá se conectar ao banco de dados onde se encontra o DW. A Figura 2 mostra um diagrama em camadas onde os principais componentes do sistema são apresentados e classificados como sendo componentes integrantes do MonaLisa ou componentes externos. É possível visualizar também alguns elementos da comunicação entre as camadas. Onde os navegadores comunicam-se com os servidores utilizando o protocolo HTTP ou HTTPS. E os servidores comunicam-se com o Sistema de Gerenciamento de Banco de Dados (SGBD) do DW utilizando linguagem SQL sobre o protocolo TCP. Os componentes externos são aqueles para os quais já existem soluções confiáveis e reconhecidas para solucionar o problema em arquiteturas. Os componentes externos são brevemente descritos abaixo:

- **Interface de rede:** disponibiliza hardware e software que permitem acesso à internet e que permite a comunicação entre os navegadores dos clientes e os servidores do MonaLisa.
- **Servidor Web:** é responsável por receber as requisições enviadas por um navegador e respondendo de acordo com o solicitado.
- **Sistema de Gerenciamento de Banco de Dados (DW):** é o sistema responsável pelo armazenamento dos dados coletados a partir dos sistemas de monitoramento de infraestrutura existentes pelo ETL e que fornecerá os dados solicitados pelo servidor web.

Figura 2 Visão geral da arquitetura em camadas da MonaLisa



Fonte: Elaborada pelo autor.

- **ETL:** é responsável por extrair, transformar e carregar o conjunto de dados ao banco de dados a partir das ferramentas de monitoramento configuradas para coleta, dentro da periodicidade determinada. Este componente é importante principalmente para viabilizar a correlação dos dados coletados dentre as diferentes ferramentas e suas variadas fontes de dados.
- **Ambiente de Monitoramento:** este é o ambiente onde temos o objeto de coleta e aplicação das diretrizes de *Software Analytics*. Cada ferramenta tem o seu conjunto de métricas e *thresholds*, que são armazenados de forma individual e não padronizada.

3.3 Modelo de Segurança

A segurança dos dados é um dos pontos críticos no desenvolvimento da ferramenta. Sendo assim, a ferramenta usa recursos de autenticação e autorização para garantir que apenas usuários devidamente autenticados tenham acesso às funcionalidades do sistema e assegurar que determinadas funcionalidades sejam acessadas por usuários autorizados. Por exemplo, observando o diagrama de casos de uso, apenas o administrador do sistema terá acesso as funcionalidades de gerenciar sistemas, gerenciar usuários e gerenciar registros. Além disso, a ferramenta também possui uma política de controle das senhas dos usuários visando definir um padrão de criação das senhas, bem como especificar quando e como eles deverão ser alteradas.

As configurações relativas à senha poderão ser realizadas pelo administrador da ferramenta. Alguns dos recursos (ou restrições) disponibilizados são: número de caracteres mínimo da senha; obrigatoriedade de caracteres numéricos e especiais; obrigatoriedade de letras minúsculas e maiúsculas; integração com programas de autenticação centralizada; e expiração da senha, período no qual o usuário não poderá usar o sistema sem trocar a senha. A ferramenta possui uma configuração padrão que pode ser modificada para atender as necessidades dos usuários da ferramenta. A configuração padrão exigida para senha será: mais de 7 caracteres, ao menos um número e um caractere especial, ao menos uma letra minúscula e maiúscula, e um tempo limite de 3 meses para atualizar a senha. Se o usuário não atualizar a senha no prazo de três meses, o mesmo não terá mais acesso ao sistema sendo exigida a atualização da senha.

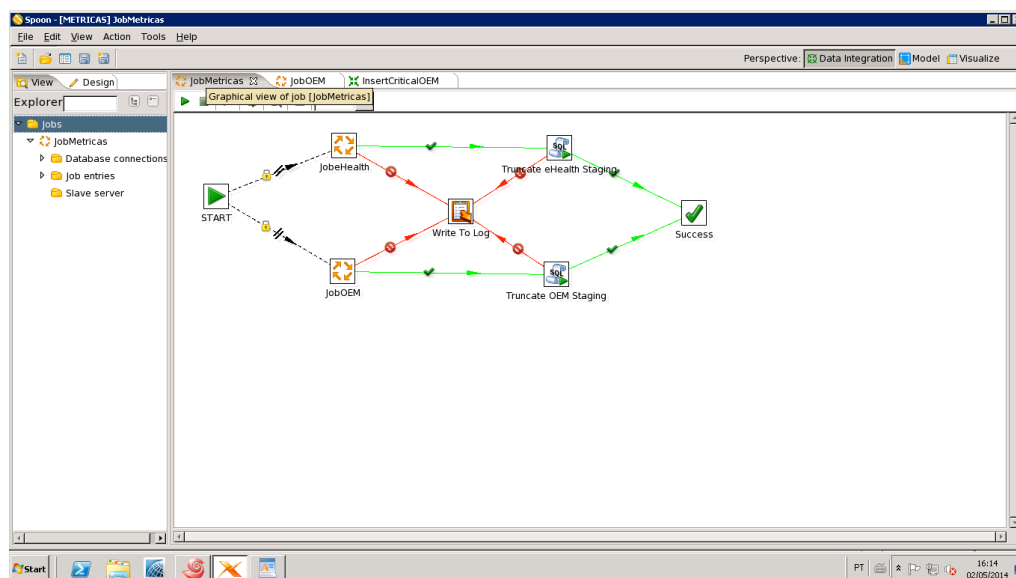
4 RESULTADOS

Esta seção tem como objetivo explicar como os requisitos da ferramenta foram implementados, apresentar as ferramentas *open source* que foram utilizadas para viabilizar implementação dos requisitos, dada a complexidade da implementação de uma ferramenta de monitoramento de infraestrutura de TI aos moldes previamente discutidos.

Após um estudo detalhado e considerando a diversidade das ferramentas de monitoramento de infraestrutura de TI, escolheu-se ferramentas open source que permitisse não só suportar a implementação dos requisitos previamente apresentados, mas também que tivesse funcionalidades que permitissem a coleta de métricas críticas para o monitoramento da infraestrutura.

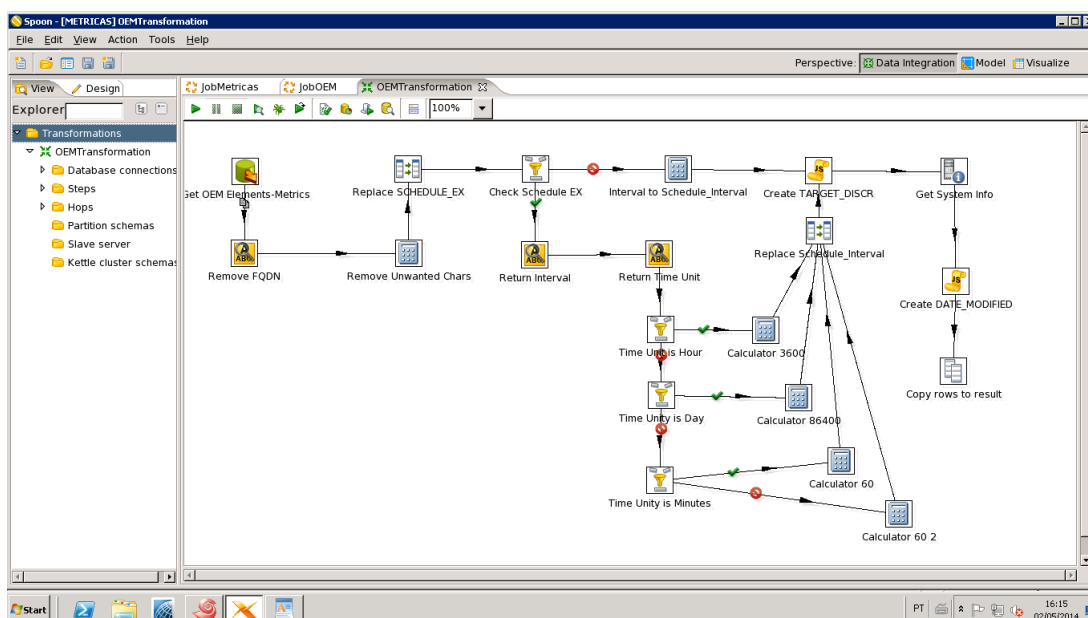
O escopo de implantação foi restrito a duas ferramentas de monitoramento de infraestrutura de TI, em um universo de centenas de ferramentas existentes no mercado. Estas ferramentas são *Oracle Enterprise Manager* (OEM) e CA eHealth, onde além de apresentarem fabricantes distintos, possuem diversos fatores diferentes tais como banco de dados (Oracle e MySQL) e sua estrutura de armazenamento de métricas e *thresholds*. Inicialmente em cada caso foi necessário mapear as tabelas onde se encontravam os dados relevantes para extração sistemática dos mesmos. Após o mapeamento foi necessário definir uma padronização para armazenamento dos dados de modo a implementar a transformação necessária em cada caso. Com isto em mãos criou-se a *query* de cada caso responsável pela efetiva extração dos dados.

Por outro lado, o *Pentaho* possui uma ferramenta gráfica para a implementação de ETL chamada *Spoon*. Nesta ferramenta foram inseridas algumas *queries* para a fase de extração dos dados, um conjunto de configurações foram definidas para permitir as transformações dos dados, bem como para permitir a carga dos dados no DW. Por padrão o *Pentaho* utiliza um banco de dados MySQL para criação do DW, mas pode ser configurado para esta utilização nos mais diversos bancos de dados através do padrão ODBC. A Figura 3 exibe a configuração do *Job* responsável por orquestrar e agendar as atividades para carga e atualização do DW.

Figura 3 Configuração do *Job* para carga do DW

Fonte: Elaborada pelo autor.

O *Spoon*, permite a configuração das atividades em níveis, onde após configurar o diagrama de orquestração se faz necessária a criação dos outros níveis. A Figura 4 apresenta no *Spoon* a configuração da transformação para o *Oracle Enterprise Manager*, que por sua vez fica dentro da atividade *jobOEM* apresentado na Figura 3.

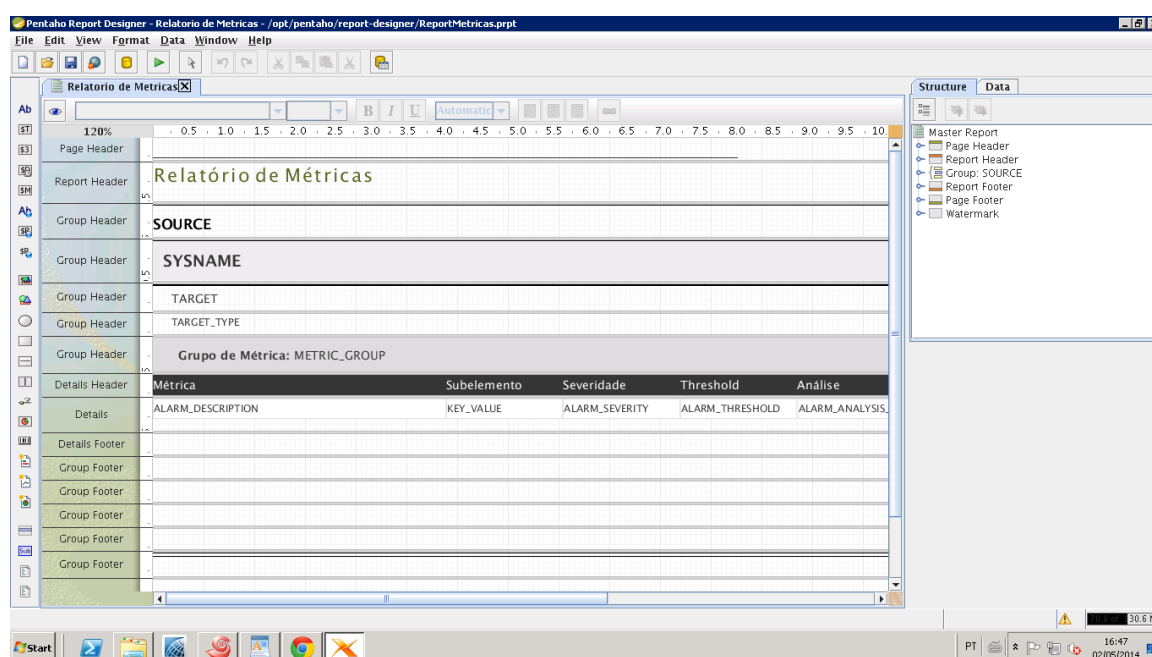
Figura 4 Configuração da transformação para o *Oracle Enterprise Manager*

Fonte: Elaborada pelo autor.

Após as transformações e a modelagem dos dados, então é executada a carga incremental dos dados no DW. Desta maneira somente os novos dados são

carregados ou alguma modificação para fins de atualização ou criação de um novo item. Após esta atividade é necessário chegar os itens que deixaram de existir para que sejam marcados como inativos no DW. De posse do DW com os dados carregados de forma coerente e com o agendamento da carga recorrente configurado, faz-se necessária a configuração do layout dos relatórios a serem fornecidos. Isto é feito na ferramenta *Pentaho Report Designer*, conforme mostrado na Figura 5.

Figura 5 Elaboração do relatório usando *Pentaho Report Designer*



Fonte: Elaborada pelo autor.

Com o relatório devidamente configurado, se torna possível através de um navegador *web* a realização de consultas e pesquisas com o modelo fornecido. Na figura abaixo é possível ver um exemplo de relatório, onde é possível verificar algumas métricas existentes para um determinado item de configuração. Cabe ressaltar que as linhas mais escuras representam os valores atuais e as mais claras (caso existam) representam valores armazenados para fins históricos. Também pode ser visualizado na figura 6 a distinção entre o tipo de severidade do alarme a ser enviado caso o *threshold* definido seja alcançado.

Figura 6 Exemplo de relatório da MonaLisa

Report Parameters

coredb1

coredb1 sicredi.net-SH
SystemEDGE UNIX System

Grupo de Métrica: Disk Partition

Métrica	Subelemento	Severidade	Threshold	Análise	Duração	Tipo
Partition Utilization	/	Major	90.00000000	300	300	Time Over Threshold
Partition Utilization	/	Critical	90.00000000	300	300	Time Over Threshold
Partition Utilization	/	Critical	95.00000000	300	300	Time Over Threshold
Partition Utilization	/admin	Major	90.00000000	300	300	Time Over Threshold
Partition Utilization	/admin	Critical	90.00000000	300	300	Time Over Threshold
Partition Utilization	/admin	Critical	95.00000000	300	300	Time Over Threshold
Partition Utilization	/dump	Major	90.00000000	300	300	Time Over Threshold
Partition Utilization	/dump	Critical	90.00000000	300	300	Time Over Threshold
Partition Utilization	/dump	Critical	95.00000000	300	300	Time Over Threshold
Partition Utilization	/varid	Major	90.00000000	300	300	Time Over Threshold

Fonte: Elaborada pelo autor.

5 TRABALHOS RELACIONADOS

Dados os objetivos descritos na Seção 1, dois tipos de trabalhos relacionados foram estudados na literatura, aqueles que lidam com os desafios de implementar um DW e aqueles que usam um DW visando produzir alguma inteligência de negócio. Dessa forma, duas ferramentas se enquadraram nesses critérios, a *Pentaho* (descrita na Seção 5.1) e a *Jasper* (analisada na Seção 5.2). Sendo assim, uma análise comparativa destas ferramentas foi realizada, visto que as mesmas são robustas e são amplamente utilizadas na prática. Além disso, as duas ferramentas possuem uma funcionalidade que é de crucial importância para proposto neste trabalho: a extração, transformação e carga de dados (ETL). Por fim, compara-se os trabalhos estudados com a ferramenta proposta (Seção 5.3).

É importante destacar que o processo de ETL visa (1) extrair dados de várias fontes externas, (2) transformar esses dados visando atender às necessidades de negócios, e (3) dar a carga de dados em um DW. Assim como todo o processo ETL, aquele disponibilizado pelo *Pentaho* e *Jasper* é capaz de consolidar dados de

diferentes fontes. Por exemplo, extraindo dados de banco de dados relacionais, arquivos de texto e dentre outros. Ter um processo ETL que seja capaz de extrair dados de diversas fontes em uma organização é de fundamental importância para promover a agregação e cruzamento dos dados visando gerar informações úteis para os tomadores de decisões. Ao utilizar *Pentaho* e *Jasper* para criar um DW, é possível armazenar informações relativas às atividades de uma organização de forma consolidada. Elas possibilitam a análise de grandes volumes de dados, os quais são coletados a partir de sistemas transacionais. Essas duas ferramentas são descritas em detalhes a seguir.

5.1 Pentaho

*Pentaho*² foi criado em 2004 e é líder em soluções *open source* de BI. Oferece uma grande variedade de recursos e através de suas soluções é capaz de manter e explorar um projeto de BI. Esta solução existe através da integração de diferentes projetos reconhecidos no mercado

5.2 Jasper

Jaspersoft³ é a empresa por trás do famoso e difundido *JasperReports*, uma solução *open source* de relatórios preferida pela maioria dos desenvolvedores para incorporar em qualquer aplicação Java que requer um sistema de notificação. A *Jaspersoft* construiu a sua própria solução de BI em torno de seu mecanismo de relatório e ele tem feito isso de uma maneira diferente do *Pentaho*. O *Jasper* tem integrado aos projetos as soluções utilizadas e conexões criadas, porém não os absorveu. Esta estratégia faz com que se dependa do *Talend* em termos de ETL e *Mondrian* para o motor OLAP. Neste ponto, é importante salientar a importância de que ambos são soluções (*Jasper* e *Pentaho*) são *open source*, mas a dependência de uma relação direta de concorrentes, como encontrado no *Pentaho*, continua a ser um fator de risco. Isto não é um problema, dada a natureza de ambos serem projetos de código aberto. *Jasper* tem código de acesso *Mondrian* e podem se adaptar e continuar a desenvolvimentos de qualquer parte do *Mondrian*

² <http://www.pentaho.com>

³ <http://www.jaspersoft.com>

5.3 Comparações entre os trabalhos estudados e o modelo proposto

Após apresentar os principais trabalhos estudados, uma comparação entre os trabalhos estudados e o modelo proposto é apresentada.

O sistema MonaLisa foi implementado como uma extensão que irá utilizar as funcionalidades existentes da ferramenta avaliada Pentaho, de forma a comprovar a utilização dos conceitos de *Software Analytics*, BI e DW para resolução dos problemas apresentados, juntamente com as vantagens também apresentadas neste trabalho.

Isso irá garantir um melhor conhecimento do assunto em si (Métricas e *Thresholds*) por parte das pessoas responsáveis pelos serviços descritos no ITIL como, por exemplo, Gestão de Capacidade e Gestão de Disponibilidade, bem como viabilizando aos tomadores de decisão um sistema único e sem a necessidade de consulta em diversos sistemas e que exigem conhecimento específico das ferramentas envolvidas. Por outro lado, o sistema desenvolvido apresenta as vantagens amplamente conhecidas de um sistema de BI, onde além dos aspectos de centralização da informação crua, também podem ser visualizados dados que sem a devida correlação não trariam maior riqueza com dados que anteriormente não eram passíveis de visualização.

Um ponto que cabe salientar são os benefícios com a nova ferramenta e que foram abordados neste trabalho. Verificou-se que nenhuma ferramenta de monitoramento conhecida armazena o histórico das informações de Métricas e *Thresholds*. Este modelo de operação inviabiliza a análise histórica de modo a trazer incertezas em casos de incidentes quanto às realizações de alterações em tais dados e que possam ter sido modificados de forma que o monitoramento tenha sido ineficaz ou ainda que o tempo de resposta ao alerta seja muito curto ou inexistente. Com a ferramenta é possível ver o histórico de alterações com as datas de quando se realizaram as alterações, bem como valores existentes anteriormente e o novo valor. Sob a ótica das modalidades do ITIL, este fato é muito relevante, visto que tais *thresholds* levam em consideração valores que garantem o pleno funcionamento da infraestrutura com performance e disponibilidade adequadas, mas ainda assim sem a existência de desperdício de recursos e consequentemente da necessidade de investimentos em equipamentos.

A ferramenta MonaLisa possui uma interface web que deve ser acessada via navegador, onde o servidor acessa um SGBD onde se encontram os dados coletados. A alimentação desta base de dados se dá por uma camada de ETL de forma agendada com periodicidade definível. Por sua vez a camada de ETL se conecta nas diferentes fontes de dados existentes nas ferramentas de monitoramento de infraestrutura de TI. Como já relatado anteriormente, tais ferramentas de monitoramento não possuem um padrão definido quanto à maneira de armazenamento dos dados, bem como inexistente um padrão quanto ao formato de armazenamento. Para isto a extração é realizada individualmente, são aplicadas as transformações necessárias aos dados para que todos façam sentido e possam ser comparadas e relacionadas entre si e entre ferramentas. Após a transformação é realizada a tarefa de cargas do SGBD, conforme já relatado anteriormente.

Uma desvantagem quanto ao modelo adotado é granularidade de coleta das informações nas ferramentas. A carga de dados em tempo real não é possível pela natureza das aplicações tanto de monitoramento como de ETL. Um agendamento com menor periodicidade, pode levar à perda de alterações realizadas em um tempo inferior ao de coleta e desta forma levando a uma perda de histórico. Um agendamento mais frequente pode trazer sobrecarga dos sistemas envolvidos e demandando um aumento desnecessário de recursos para uma tarefa considerada de retaguarda operacional.

6 CONSIDERAÇÕES FINAIS

Com o presente trabalho foi possível entender e implementar os principais conceitos de *Software Analytics*. Conhecer pesquisas atuais sobre o assunto e sistemas que objetivam solucionar problemas relacionados ao fato da existência de diversas fontes distintas de dados sem um local único de consulta através de técnicas já comprovadamente efetivas. Através do entendimento dos conceitos e aplicabilidade de *Software Analytics*, *Data Warehouse* e BI foi possível entender melhor a maneira correta do cenário pretendido e situações similares com a aplicação dos conceitos de forma efetiva para posterior implementação prática.

Os estudos relacionados ao monitoramento de infraestrutura de TI, em especial quanto a existência de um gerenciador central mas sem a existência de um

banco de dados de métricas e *thresholds*, possibilitaram a correlação e posterior utilização dos conceitos aprendidos para a resolução de um problema que é encontrado nos mais diversos domínios da TI: a existência de diversas fontes de dados relevantes e com a impossibilidade de extração de informações úteis e algumas desconhecidas sem uma visualização unificada.

O conhecimento acima descrito foi essencial para a criação do sistema MonaLisa. O sistema foi desenvolvido observando as boas práticas de Engenharia de Software no que se refere as etapas do processo de desenvolvimento de software, incluindo a análise do problema, especificação de requisitos, modelagem da arquitetura do sistema, implementação e integração, testes e validação, implantação e manutenção. Durante a modelagem e implementação a utilização de UML, bancos de dados relacional e serviços web foram importantes para colocar em prática muitos dos conceitos aprendidos durante o curso de graduação.

5.1 Trabalhos Futuros

Tendo em vista a complexidade e diversidade de ferramentas de monitoramento de infraestrutura de TI, foi escolhida uma ferramenta de código aberto para implementação e comprovação de resolução do problema proposto que possuísse funcionalidades que satisfizessem os requisitos descritos por *Software Analytics*. O escopo de implantação foi restrito a duas ferramentas de monitoramento de infraestrutura de TI, em um universo de centenas de ferramentas existentes no mercado. Porém, o presente trabalho pode ser continuado e podem ser incluídas novas ferramentas de monitoramento de infraestrutura de TI não mapeadas, com a finalidade de enriquecer e cobrir um percentual maior de ambientes, métricas e *thresholds*. Também é possível a criação de uma ferramenta específica que contenha funcionalidades desejáveis para a resolução do problema, desvinculando da ferramenta *Pentaho* e ampliação de funcionalidades desejáveis quanto ao domínio utilizado.

Algumas sugestões para o desenvolvimento futuro deste trabalho são:

- Melhorar as funcionalidades da ferramenta, criar a possibilidade de desenvolvimento de plug-ins para cada nova ferramenta de monitoramento de infraestrutura de TI, de forma a automatizar a

implantação de novas coletar e minimizar erros provenientes de interpretações que se façam necessárias;

- Evoluir no modelo de normalização de dados de métricas e *thresholds* provenientes de ferramentas de monitoramento de TI, visto que ao realizar a coleta de dados, se faz necessário o desenvolvimento do formato para que os dados se tornem coerentes após a sua transformação;
- Criar uma interface para dispositivos móveis, pois o MonaLisa será utilizado principalmente em meio corporativo, onde atualmente a existência e utilização de dispositivos móveis, *tablets* e *smartphones* é bastante difundida. Esta atividade inclusive encoraja e apoia o uso da ferramenta entre os tomadores de decisão de grandes empresas que fazem grande uso de tais dispositivos.

7 REFERÊNCIAS

BUDGEN, D.; BURN, A J.; BRERETON, O. P.; KITCHENHAM, B. A; PRETORIUS, R. **Empirical evidence about the UML: a systematic literature review**. Software Practice and Experience, v. 41, n. September 2010, p. 363–392, 2011. Wiley Online Library. Disponível em:

<<http://onlinelibrary.wiley.com/doi/10.1002/spe.1009/abstract>>

BUSE, R.; ZIMMERMANN, T. **Information Needs for Software Development Analytics**, Proc. International Conference on Software Engineering (ICSE), IEEE CS, 2012.

CRUZ-LEMUS, J. A; GENERO, M.; CAIVANO, D.; et al. **Assessing the influence of stereotypes on the comprehension of UML sequence diagrams: A family of experiments**. Information and Software Technology, v. 53, n. 12, p. 1391–1403, 2011. Disponível em: <<http://www.sciencedirect.com/science/article/pii/>>. Acesso em: 17/8/2013.

DAVENPORT, H.; HARRIS, J.; MORISON, R. **Analytics at Work: Smarter Decisions**, Better Results, Harvard Business Review Press, 2010.

DRAGICH L. **APM and MoM – Symbiotic Solution Sets**. Disponível em <http://apmdigest.com/>, 2013

DZIDEK, W. J.; ARISHOLM, E.; BRIAND, L. C. **A Realistic Empirical Evaluation of the Costs and Benefits of UML in Software Maintenance**. IEEE Transactions on

Software Engineering, v. 34, n. 3, p. 407–432, 2008. IEEE Press. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4459340>>.

FARIAS, K.; GARCIA, A.; WHITTLE, J.; CHAVEZ, C.; LUCENA, C. **Evaluating the Effort of Composing Design Models : A Controlled Experiment** ,2011.

KITCHENHAM, B.; PRETORIUS, R.; CRUZ-LEMUS, D.; et al. **Systematic literature reviews in software engineering – A tertiary study**. Information and Software Technology, v. 52, n. 8, p. 792–805, 2010. Elsevier B.V. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0950584910000467>>. Acesso em: 13/8/2013.

LUCAS, F. J.; MOLINA, F.; TOVAL, A. **A systematic review of UML model consistency management**. Information and Software Technology, v. 51, n. 12, p. 1631–1645, 2009. Elsevier B.V. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0950584909000433>>. Acesso em: 17/8/2013.

MENS, T. **A State-of-the-Art Survey on Software Merging**. , v. 28, n. 5, p. 449–462, 2002.

MENZIES, T.; SHULL, F. **Empirical Software Engineering**, tech. briefing, Proc. International Conference on Software Engineering, IEEE CS, 2011.

MENZIES, T.; ZIMMERMANN, T. **Software Analytics: So What?**, IEEE Software, July/Agust, 2013

MENZIES T. ET AL. **The Inductive Software Engineering Manifesto: Principles for Industrial Data Mining**. Disponível em <<http://menzies.us/pdf/11manifesto.pdf>>, Proc. Int'l Workshop Machine Learning Technologies in Software Eng., 2011.

MILES, R.; HAMILTON, K. **Learning UML 2.0**. [S.l.]: O'Reilly Media, 2008.

OBJECT MANAGEMENT GROUP. **Unified Modeling Language**. Disponível em: <<http://www.omg.org/spec/UML/>>. Acesso em 13 de ago. de 2013.

PORTER, A.; SELBY, R. **Empirically Guided Software Development Using Metric-Based Classification Trees**, IEEE Software, vol. 7, no. 2, 1990, pp. 46–54.

WOHLIN, C.; RUNESON, P.; HÖST, M.; OHLSSON, M.; REGNELL, B.; WESSLÉN, A. **Experimentation Software Engineering - An Introduction**, Kluwer Academic Publishers, 2000.

ZHANG, D.; HAN, S.; DANG, Y.; LOU, J.; ZHANG, H. **Software Analytics in Practice**. , 2012.