http://dx.doi.org/10.17058/tecnolog.v20i1.6569

TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

# CONSIDERAÇÕES PARA IMPLEMENTAÇÃO DE FERRAMENTAS MULTIPLATAFORMA PARA MONITORAMENTO DE SISTEMAS VIRTUALIZADOS

*Fillipi de P. Suszek[1], Ricardo M. Czekster[1], Thais Webber[1], César Marcon[2], Rodrigo Nedel[1]**

*[1] Departamento de Informática – Universidade de Santa Cruz do Sul (UNISC)*
*Avenida Independência, 2293 – CEP 96815-900 – Santa Cruz do Sul – RS – Brasil*

*[2] Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)*
*Avenida Ipiranga, 6681 – CEP 90619-900 – Porto Alegre – RS – Brasil*

**E-mail: ricardoc@unisc.br*

*Recebido em: 27/09/2015*
*Aceito em: 25/11/2015*

**RESUMO**

A virtualização através do uso de máquinas virtuais sobre máquinas físicas, para executar diferentes sistemas em diferentes domínios de aplicação tem sido uma abordagem comumente adotada em diferentes contextos. A possibilidade de se abstrair plataformas, infraestrutura ou software como um serviço passou a ser uma técnica válida para executar sistemas usando Computação na Nuvem (*Cloud Computing*) onde a virtualização é uma das principais tecnologias para efetivar sua utilização. O uso de ambientes virtualizados é determinante, entretanto, em muitos casos, estas tecnologias são escolhidas sem levar em conta o desempenho ou outros atributos não funcionais, tais como a garantia de qualidade de serviço (QoS), resiliência, confiabilidade, tolerância a falhas e escalabilidade, para citar algumas. O objetivo deste trabalho foca nas considerações principais para concepção de uma ferramenta completa de monitoramento e inspeção de sistemas virtualizados. A ideia é poder estimar as melhores configurações de software e hardware para plataformas virtualizadas sem que ocorra degradação de desempenho. Para demonstrar a efetividade da técnica foi implementada como exemplo uma ferramenta de propósito específico de monitoramento chamada VM-MON, descrita no presente trabalho.

**Palavras-chave:** virtualização, implementação de sistemas, monitoramento.

## 1 Introdução

O aumento da complexidade das aplicações em termos de implementação, adoção de tecnologias de Computação na Nuvem (*Cloud Computing*) e questões gerais ao escolher a infraestrutura de execução das aplicações torna a avaliação de desempenho um quesito primordial para a análise quantitativa para uso racional dos recursos existentes. É através desta análise que se inferem índices de desempenho que atestam a eficácia e a utilização de recursos de diferentes sistemas (computacionais ou não). Nos últimos anos, observou-se o uso massivo e indiscriminado de soluções virtualizadas com pouca ou nenhuma preocupação sobre a qualidade da execução e outros requisitos não-funcionais, tais como tolerância a falhas ou planejamento de capacidade.

Quando se fala em desenvolvimento de software, diversas fases estão previstas, destacando-se a fase de validação e verificação (V&V) onde um dado sistema é comparado frente a sua especificação funcional onde, conforme o objetivo do estudo atesta-se desempenho e procuram-se defeitos [1]. Em linhas gerais, caso defeitos sejam encontrados (o que normalmente ocorre, invariavelmente, para muitos sistemas), a fase de implementação e projeto é reiniciada, onde o produto final possui uma série de ciclos dependendo da metodologia de desenvolvimento de software escolhida. O objetivo da V&V é inspecionar um sistema quanto aos seus Requisitos Funcionais, ou seja, estabelece um contrato do que um dado software deve implementar como funcionalidades [2,3]. Em muitas organizações, a fase de verificação funcional é a etapa mais importante para verificar a qualidade final de um determinado produto, entretanto, pouca atenção é dada para os aspectos *não-funcionais* de desempenho, também denominados aspectos qualitativos. Estas propriedades dizem respeito à qualidade dos sistemas, aos cálculos dos tempos de resposta médio das aplicações, de usabilidade, segurança, identificação dos principais gargalos de desempenho conforme a carga submetida, entre outras.

Observa-se comumente a escolha indiscriminada de técnicas de virtualização sem a devida preocupação com os requisitos não-funcionais. Este problema muitas vezes só é evidenciado quando o sistema encontra-se em produção, tendo recebido uma carga de trabalho (*workload*) excessiva ou não prevista em uma especificação original. Em virtude da utilização destas novas tecnologias deve-se atentar para o fato que os sistemas devem ser testados sob o ponto de vista do desempenho observando-se o incremento do número de

http://dx.doi.org/10.17058/tecnolog.v20i1.6569
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

usuários, a escalabilidade do sistema e a disponibilidade dos recursos que compõem os sistemas. Deste modo, novas práticas e técnicas são necessárias para estimar e medir sistemas computacionais onde a virtualização é um pré-requisito para a execução de um sistema.

A vantagem do uso de monitoramentos de recursos chave à operação dos sistemas tange a obtenção de índices de desempenho que atestam sua operação básica e são fundamentais para entender gargalos e problemas variados de sistemas computacionais. Os analistas utilizam estes dados como parâmetros para a modelagem analítica, permitindo a criação de múltiplos cenários de análise com a variação dos parâmetros, uma alternativa barata para estudar diferentes realidades.

O objetivo deste trabalho, então, é uma tentativa de mitigar estes problemas através da concepção de uma ferramenta chamada VM-MON para monitorar e inspecionar sistemas virtualizados. De forma complementar, o presente artigo visa o estudo de tais configurações para detectar, estimar e prever degradações de desempenho de sistemas virtualizados dada a carga bem como explorar aspectos de escalabilidade.

## 2 Avaliação de Desempenho de Sistemas

A seguir são apresentados conceitos de Avaliação de Desempenho e virtualização, detalhando os principais objetivos de cada técnica e como serão utilizados para mapear sistemas virtualizados em modelos analíticos.

### *2.1 Técnicas de avaliação e modelagem*

A modelagem computacional de sistemas é uma parte importante da avaliação analítica, pois é responsável por transformar descrições abstratas em primitivas que evidenciam a operação e a qualidade do sistema sob estudo (*System Under Study* – SUT). A necessidade de métodos formais para avaliação computacional de desempenho é importante para explorar raciocínios (*reasoning*) em sistemas e auxiliar na descoberta de gargalos e problemas operacionais antes da sua implementação física [4,5,6].

O objetivo de modelar sistemas é formalizar uma realidade com premissas lógicas de interação e comportamento e permitir a construção de modelos que representassem realidades com o maior número de detalhes possível, baseando-se em primitivas de modelagem. A próxima etapa, após a concepção de um modelo representativo, é o cálculo de índices para evidenciar as propriedades estacionárias do sistema e estudando a ocorrência de um regime de equilíbrio nas probabilidades de permanência de cada estado. O mapeamento de diferentes realidades para modelos permite a determinação precisa das interconexões das entidades envolvidas, capturando as suas nuances e respondendo questões críticas, analisadas a partir dos índices de desempenho calculados como utilização dos recursos ou tempo de resposta.

A próxima etapa é a solução do modelo e busca o equilíbrio do sistema onde este é analiticamente resolvido através de cálculos que inferem índices de desempenho. O objetivo da fase de solução é calcular as probabilidades de permanência dos estados modelados e analisar seu impacto em diferentes versões de modelos que foram concebidas para a realidade sendo estudada. Para resolver modelos deste tipo, calcula-se o vetor de probabilidades da estacionariedade do sistema, ou seja, supõe-se que este foi 'simulado' até um ponto que as mudanças não mais afetam o estado inicial. Pode-se dizer que o sistema chegou a um equilíbrio, dadas as taxas definidas para as transições. Assim, extraem-se os índices de desempenho [4].

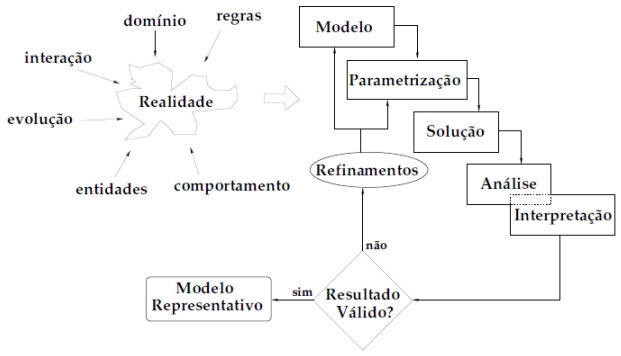


Figura 1 – Processo geral de modelagem de sistemas [6].

A Figura 1 representa o processo de modelagem de sistemas para avaliação de desempenho, desde a captura da operação fundamental até a verificação da validade dos modelos construídos. Para realizar avaliações de desempenho de sistemas existem três técnicas mais tradicionais: o monitoramento, a simulação e os métodos analíticos [7,8,9]. A técnica da modelagem analítica iniciou com a definição das Cadeias de Markov (CM), onde os componentes são apresentados na forma de um grafo contendo as transições entre os estados que um modelo assume. A partir dos anos 80 surgiram os formalismos estruturados baseados em Cadeias de Markov [10], onde os maiores exemplos são as Redes de Filas de Espera (*Queueing Networks* – QN) [4,9], as Redes de Petri – do inglês *Petri Nets* – (PN) [11,12], as Redes de Petri Coloridas (*Coloured Petri Nets* – CPN) e Álgebra de Processos (*Stochastic Process Algebra* – SPA) [13].

Já a técnica do monitoramento, foco do presente trabalho, necessita do sistema em funcionamento para que seja possível sua utilização, bem seguir processos pré-determinados para que as medidas efetuadas sejam confiáveis. Esta técnica é interessante quando se têm o sistema sob análise disponível para ser executado inúmeras vezes, já que é o conjunto de monitoramentos mais uma análise estatística quem dita os principais índices produzidos. O problema do monitoramento é

http://dx.doi.org/10.17058/tecnolog.v20i1.6569
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

a sua disponibilidade irrestrita; muitas vezes não é possível se monitorar um sistema de forma consistente.

Outro problema que normalmente se constata nesta técnica é que o efeito de se monitorar um sistema, ou seja, instrumentá-lo (colocar diretivas de manipulação e captura de tempos, por exemplo) influencia no gasto de tempo, o que altera negativamente as amostras produzidas (este é o '*Paradoxo do Monitoramento*'). O ideal é usar a experiência para monitorar um sistema de forma consciente de forma que não impacte negativamente nas medidas capturadas. Neste sentido, sistemas virtualizados, de alto desempenho e computacionalmente intensivos são particularmente difíceis de serem monitorados, devido aos problemas acima mencionados, o que por outro lado se constitui em um problema interessante de pesquisa a ser estudado.

*2.2 Virtualização*

A virtualização é uma técnica que permite a separação entre as aplicações e o sistema operacional (SO) de componentes físicos [14,15,16]. Ao se criar uma determinada Máquina Virtual (*Virtual Machine*), esta possui uma determinada aplicação sendo executada em um determinado SO, convenientemente escolhido para executar o sistema computacional. É possível criar uma série de máquinas virtuais em um mesmo hardware, cada uma explorando uma determinada característica do SO escolhido, por exemplo, é possível instalar diferentes versões do SO da Microsoft, como o Windows™ e/ou GNU/Linux de diferentes distribuições (Ubuntu, Lubuntu, Fedora, Mint, RedHat, etc). O objetivo ao se utilizar a virtualização reflete uma substancial redução de custos e complexidade.

Existem diversas vantagens ao se virtualizar aplicações. No contexto de teste de software, por exemplo, pode-se virtualizar diferentes plataformas de computação para verificação de funcionalidades e diferentes configurações em ambientes de teste. É possível construir um ambiente contido que opera apenas no seu contexto de rede, isolado de outros problemas que talvez fossem acontecer (por exemplo, queda de servidores, etc).

Outra característica interessante é permitir aumento sob demanda de recursos físicos (i.e., capacidade elástica de provisionamento) conforme o número de transações ou o número de usuários aumentem sem interferência humana. Esta característica também implica em questões de computação verde (*Green Computing*), e questões de precificação (*pricing*) de clusters de computadores onde os clientes compram recursos computacionais conforme diferentes modelos e opções. Estas características tornam a adoção de virtualização cada vez mais automatizadas e práticas do ponto de vista operacional, pois coordenam a criação e instanciação de recursos conforme o uso. O fato de permitir que os usuários escolham as plataformas em tempo de execução dos sistemas confere um alto grau de flexibilidade a esta solução virtualizada.

No contexto deste artigo, foca-se na virtualização de servidores (*Server Virtualization*) uma vez que se deseja trabalhar com diversas máquinas virtuais que são executadas em cima de um sistema operacional baseado na plataforma Microsoft Windows (MS-Windows), GNU/Linux ou outra (e.g. Unix). Os principais exemplos de softwares que permitem a instalação e configuração de máquinas virtuais (hipervisores) são: vSphere™ (VMware), XenServer (Citrix), Hyper-V™ (Microsoft), VirtualBox™ (Oracle) e outras técnicas baseadas em *Kernel-based Virtual Machine* (KVM).

*2.3. Particularidades de monitoramento de sistemas virtualizados*

Esta pesquisa possui diversos desafios operacionais. Por exemplo, as opções escolhidas de cada gerenciador de máquina virtual pode impactar bastante no tempo de execução das aplicações virtualizadas. O interessante é detectar e predizer, ao se construir o ambiente virtualizado, boas configurações bem como a carga (em termos de transações e usuários) máxima sem perda de desempenho e sem comprometer a solução.

A virtualização utiliza uma entidade que supervisiona o acesso aos principais recursos, esta chamada de hipervisor (*hypervisor*), ou seja, uma camada de software que controla o hardware do sistema e oferece uma visão abstrata do hardware para acesso. O uso de sistemas virtualizados normalmente está relacionado não apenas com as opções de virtualização bem como com aspectos avançados tais como alocação dinâmica de recursos, provisionamento de mais máquinas virtuais para garantir uma possível demanda futura, a migração de máquinas virtuais em uma rede, os aspectos de segurança e o monitoramento de recursos críticos à sua operação que visem à detecção de desgastes de desempenho.

O sistema de monitoramento implementado tendo-se em vista a utilização em sistemas virtualizados é multiplataforma, obrigatoriamente, pois precisa ser executado em diferentes SOs. A ideia é que o sistema de monitoramento descubra o SO onde está executando e derive os comandos específicos para esta plataforma, em tempo de execução. Sistemas de monitoramento em plataformas virtualizadas também devem ser extremamente fáceis de serem utilizados, possuindo alta amigabilidade ao usuário final (*user friendliness*), já que se pressupõe que este não possui os conhecimentos técnicos para realizar um monitoramento formal, não possui licenças de ferramentas já existentes devido ao alto custo de aquisição e deseja implantar em sua organização uma solução virtualizada.

Cabe ressaltar que o monitoramento de máquinas virtuais pode ser realizado por ferramentas tradicionais e já estabelecidas de redes de computadores que também

http://dx.doi.org/10.17058/tecnolog.v20i1.6569
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

monitoram os recursos das redes e das máquinas físicas, tais como Nagios, MRTG, NTOP, Wireshark, etc, sem perda de generalidade. Estas ferramentas são muitas vezes utilizadas para monitorar recursos de redes distribuídas, utilizando protocolos de comunicação padrão como *Simple Network Management Protocol*, ou SNMP [17]. Entretanto, também podem ser utilizadas ferramentas específicas da máquina virtual que foi instalada, por exemplo, ferramentas básicas de monitoramento de memória, CPU, discos rígidos e rede tais como `iostat`, `vmstat`, `ps` e `netstat` (no GNU/Linux), explicadas na Seção 3.1, entre outras, bem como a utilização de contadores de desempenho (no MS-Windows).

Existem diversas ferramentas proprietárias para analisar o desempenho de máquinas virtuais. Os problemas destas abordagens é a viabilidade econômica para aquisição de licenças. Exemplos de ferramentas que podem ser citadas são específicas dos hipervisores dos softwares proprietários do Hyper-V, da vSphere ou do VirtualBox, entre outras (dependendo da ferramenta que foi escolhida pelos usuários).

A proposta deste trabalho é utilizar as ferramentas gratuitas já existentes nos SOs, provendo uma interface gráfica amigável para usuários que não estão acostumados a realizar tarefas administrativas e que desejam utilizar soluções de virtualização e Computação na Nuvem nas suas organizações.

### *2.4 Trabalhos relacionados*

A comunidade científica retornou o interesse à técnica da avaliação de desempenho de sistemas através de monitoramento com a adoção cada vez mais ampla de virtualização pelas organizações. Existem muitos trabalhos recentes que discutem estas possibilidades de pesquisa com resultados interessantes. Por exemplo, os autores [18,19,20] apresentam *surveys* interessantes sobre o assunto do monitoramento aplicado à computação na nuvem.

Existem diversas ferramentas de monitoramento para computação na nuvem, podendo-se citar os trabalhos de [21,22,23] que, respectivamente, apresentam ferramentas tais como PAPI-V, GMonE e DARGOS, implementadas especificamente para tratar do problema de monitoramento permanente de sistemas virtualizados. Outro trabalho relacionado interessante e próximo do descrito neste artigo é a ferramenta VEPmon [24] que implementou um sistema de monitoramento de ambientes virtualizados para o hipervisor Citrix Xen, descrito inicialmente em [14].

Os autores [25] se preocuparam em divisões (que os autores denominaram *slices*) de monitoramento aplicados à nuvem. Análises comparativas, casos de uso, monitoramentos de recursos na nuvem e frameworks de ferramentas de monitoramento são apresentadas em [26,27,28]. Estes trabalhos são extremamente atuais e reflete a preocupação a comunidade científica em propor ferramentas para tratar com o tópico do monitoramento de sistemas na nuvem em contextos virtualizados.

## 3 Considerações para implementação de ferramentas

A seguir, apresenta-se uma ferramenta que implementa diversas considerações para trabalhar com monitoramento de sistemas virtualizados. Para exemplificar os conceitos vistos neste trabalho, foi implementada uma ferramenta de apoio que mostra as considerações mais importantes de codificação de soluções de monitoramento virtualizado. A ferramenta foi chamada de VM-MON e foi desenvolvida na Linguagem de Programação Java™, devido às características multiplataforma oferecidas além de ser uma linguagem amplamente utilizada para o desenvolvimento de sistemas de pequeno à grande porte. A ideia é que a VM-MON seja executada em qualquer SO, entretanto, nesta versão, o foco é dado para máquinas GNU/Linux e MS-Windows. Versões futuras da ferramenta poderão implementar diretivas de monitoramento para FreeBSD, Unix, entre outros SOs. O hipervisor escolhido para executar a VM-MON foi o Virtual Box, da Oracle, devido à sua gratuidade e facilidade de instalação e operação de vários SOs.

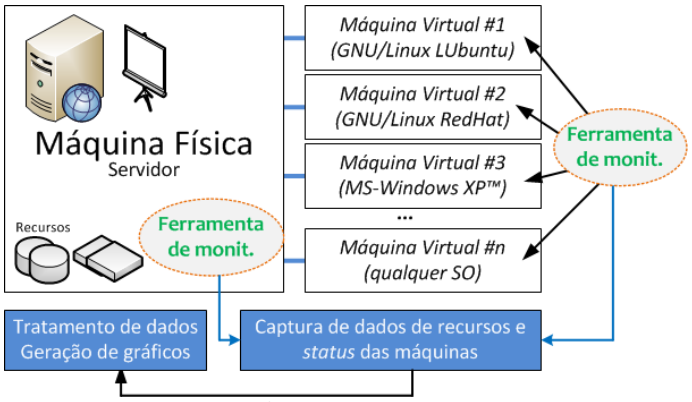


Figura 2 – Execução de ferramentas de monitoramento em contextos virtualizados.

A Figura 2 mostra os elementos monitorados por ferramentas de monitoramento em contextos físicos e virtualizados. Cabe ressaltar que a ferramenta executa múltiplas instâncias (uma na máquina física e uma em cada máquina virtual). A ferramenta demonstrada neste artigo apresenta uma GUI (*Graphical User Interface*) para comandos usualmente não triviais de monitoramento para plataformas GNU/Linux e MS-Windows que pode ser executada para monitorar quaisquer sistemas virtualizados a partir destes SOs. Cabe ressaltar que a ferramenta é amigável ao usuário e possui opções para escolher os dispositivos a serem monitorados bem como auxilia os usuários a escolher a quantidade de tempo de monitoramento desejado. Uma vez que estes dados são escolhidos, a ferramenta gera scripts para os SOs específicos contendo os comandos para as ferramentas auxiliares (descritas

http://dx.doi.org/10.17058/tecnolog.v20i1.6569
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

na próxima seção) serem executadas. Uma vez que o tempo escolhido pelo usuário chega ao seu término, o sistema permite que o usuário inicie o processo de análise dos dados, executando scripts para transformar dados brutos em dados legíveis pelas ferramentas gráficas (no caso, estamos utilizando Perl e gnuplot para estes fins específicos). Por fim, a ferramenta permite que uma nova sessão de análise seja feita pelo usuário, já que organiza os arquivos em pastas geradas a partir do tempo (formato hh:mm:ss), salvando os dados para análises históricas.

*3.1 Dispositivos de monitoramento e ferramentas auxiliares*

Os dispositivos escolhidos para efeitos de monitoramento pela VM-MON são: a memória principal (*memory*), a CPU (*Central Processing Unit*), o HD (*Hard Disk*) e as interfaces de comunicação em rede (Ethernet Interfaces). No GNU/Linux, foram escolhidas as ferramentas auxiliares de monitoramento: ps (*process snapshot*), vmstat (*virtual memory statistics*), iostat *(input/output statistics*) e netstat (*network statistics*). A ferramenta utiliza diversos parâmetros de cada ferramenta auxiliar para extrair os dados para análise. No MS-Windows, a ferramenta adota os contadores de desempenho (*performance counters*) através de chamadas específicas de linha de comando específicas. Para processar os dados brutos gerados por cada ferramenta auxiliar foi utilizada a Linguagem de *Scripting* Perl (*Practical Extraction and Report Language*), e para construir os gráficos para análise utilizou-se a ferramenta da biblioteca GNU gnuplot. Cabe ressaltar que tanto Perl quanto gnuplot são ferramentas compiladas para GNU/Linux e MS-Windows, não necessitando de quaisquer conversões e recompilações. A Tabela 1 a seguir indica os principais contadores de desempenho utilizados para monitorar as máquinas MS-Windows.

Tabela 1 – Listagem dos contadores de desempenho do MS-Windows.

| **Arquivo de contadores de desempenho: `perfconfig.txt`** |
|---|
| `"\Memory\Available Bytes"` |
| `"\Memory\Pages/Sec"` |
| `"\Server\Bytes Total/Sec"` |
| `"\Paging File(\??\C:\pagefile.sys)\% Usage"` |
| `"\PhysicalDisk(_Total)\% Disk Time"` |
| `"\PhysicalDisk(_Total)\Avg. Disk Queue Length"` |
| `"\PhysicalDisk(_Total)\Disk sec/Transfer"` |
| `"\Process(java)\Working set"` |
| `"\Process(javaw)\Working set"` |
| `"\Process(eclipse)\Working set"` |
| `"\Processor(_Total)\% Processor Time"` |
| `"\Processor(_Total)\Interrupts/sec"` |
| `"\Thread(_Total/_Total)\Context Switches/sec"` |
| `"\System\Processor Queue Length"` |

Basicamente, os contadores criados existem para observar o tempo do disco (*Disk Time*), o tamanho médio da fila dos processos esperando o disco (*Avg. Disk Queue Length*), o Conjunto de Trabalho (*Working Set*) de alguns processos (java, javaw e eclipse), o tempo do processador (*Processor Time*), o total de interrupções por segundo (*Interrupts/sec*), as trocas de contexto por segundo (*Context Switches/sec*) e o tamanho da fila do processador (*Processor Queue Length*). Cabe lembrar que outros contadores de desempenho podem estar presentes, dependendo das opções dos usuários, sendo estes apresentados alguns dos mais importantes das máquinas com MS-Windows.

*3.2 Detalhes de implementação*

A VM-MON utiliza o artifício de executar comandos independentes de plataforma de Java sempre que possível (e.g., para criar um diretório, pois este processo depende do SO e Java possui implementações multiplataforma destinadas a este propósito) em conjunção com chamadas de sistema (que executam os scripts particulares de cada SO).

A ferramenta emprega diversas linguagens e ferramentas auxiliares em um único ambiente de execução, por exemplo, utiliza no GNU/Linux *scripts* bash (.sh), enquanto que no MS-Windows, utiliza arquivos de lote (.bat). A ferramenta processa os dados em Perl, gerando arquivos com os resultados tabulados em colunas que representam os dados requisitados pelos usuários e então dispara comandos para gerar gráficos e permitir a análise. A VM-MON foi desenvolvida utilizando-se a biblioteca gráfica Java Swing que fornece uma API (*Application Programming Interface*) rápida, responsiva e fácil de programar e estender para a adição de funcionalidades futuras.

O objetivo da ferramenta é poder ser executada em qualquer contexto virtualizado. Por exemplo, se o usuário possuir plataformas com MS-Windows instalado (como máquina física principal) e múltiplos SOs virtualizados com GNU/Linux, a ferramenta, se executada em ambas as plataformas, poderá monitorar os recursos da máquina física bem como os recursos de cada máquina virtual em execução. O usuário pode escolher diferentes dispositivos para monitorar em diferentes contextos de execução, tanto físicos quanto virtualizados, o que demonstra a importância de uma ferramenta multiplataforma voltada à análise de desempenho de sistemas deste porte.

*3.3 Principais comandos e opções de ferramentas*

Conforme mencionado anteriormente, a VM-MON realiza a chamada de diversas outras ferramentas auxiliares provendo uma interface gráfica amigável. A Tabela 2 a seguir mostra os SOs disponíveis e as chamadas de sistema que são

http://dx.doi.org/10.17058/tecnolog.v20i1.6569
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

realizadas internamente pela VM-MON para capturar resultados de monitoramento nas duas plataformas.

Tabela 2. Comandos disparados internamente na VM-MON com chamadas de sistema.

| *Sistema Operacional* | *Ferramenta* | *Comando de disparo da ferramenta auxiliar* |
|---|---|---|
| **MS-Windows** | **logman** | `logman create counter dataCollection -s SERVER1 -cf perfconfig.txt -o \\path\perfmon\%COMPUTERNAME%_dataCollection -v ddMMhhmm -f csv -si 10 -u admin admin` |
| | | `logman start dataCollection -s SERVER1 -u admin admin` |
| | | `logman stop dataCollection -s SERVER1 -u admin admin` |
| | | `logman delete dataCollection -s SERVER1 -u admin admin` |
| **GNU/Linux*** | **iostat** | `iostat -x $sleeptime $iter >> io.dat` |
| | **vmstat** | `vmstat $sleeptime $iter >> vm.dat` |
| | **ps** | `ps -eo"pcpu,pmem,time,comm" \| sort -r \| head -11 \| tail -n 10 >> ps.dat` |
| | **netstat** | `netstat -i \| grep eth >> neth.dat` |

*(*) Estes são **exemplos** de comandos possíveis. As variáveis $sleeptime, $iter vem das opções dos usuários.*

A ferramenta logman normalmente vem instalada juntamente com o MS-Windows (com a perfmon.exe). No GNU/Linux necessitam da instalação do pacote sysstat (*System Statistics*), já para o MS-Windows é preciso criar os contadores de desempenho via scripts de inicialização (.bat) ou MS-Windows PowerShell, que criam os contadores (*create counter*) usando a lista de contadores de perfconfig.txt (olhar Tabela 1), com formato de arquivo específico (DiaMêsHoraMin, ou ddMMhhmm), tipo (*comma separated values*, ou CSV), para um servidor específico (SERVER1), com intervalo de monitoramento e um usuário e senha (-u). Também é possível começar um monitoramento (start), parar (stop) e excluir uma coleção de dados monitorados (*delete*). No GNU/Linux, existe um laço que só termina quando o tempo de monitoramento especificado pelo usuário acaba. Cabe ressaltar que um arquivo por ferramenta auxiliar é sempre criado (arquivo .dat) em uma pasta específica e a cada iteração do laço novos dados são adicionados (*appended*).

### *3.4. Execução da VM-MON*

A seguir são listadas algumas telas da ferramenta bem como sua execução em um contexto virtualizado baseado em uma máquina que está executando o GNU/Linux com a distribuição Lubuntu (uma versão mais leve e rápida do Ubuntu).

A ideia da VM-MON é ser leve (*lightweight*), rápida e fácil de configurar os monitoramentos necessários. Para que a ferramenta seja utilizada por uma quantidade considerável de usuários, decidiu-se que sua interface seria implementada em inglês. A Figura 3 mostra algumas telas principais da ferramenta.

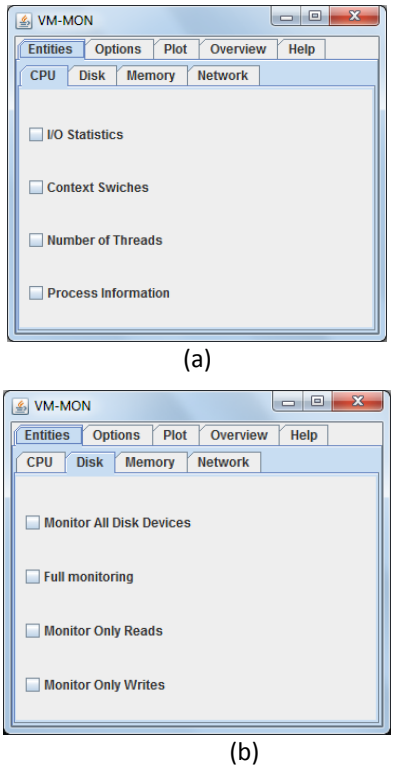


Figura 3 – Tela principal da ferramenta VM-MON, mostrando a aba '*Entities*' e os monitoramentos da CPU em (a) e monitoramentos no disco rígido em (b).

A Figura 3 mostra a VM-MON por completo, onde é possível verificar diversas abas que o usuário pode configurar monitoramentos (aba *Entity*), configurar opções (aba *Options*), gerar gráficos automaticamente (aba *Plot*), resumo das escolhas (aba Overview) e ajuda do sistema (aba *Help*). A Figura 3a mostra algumas opções de monitoramento de disco tais como estatísticas gerais de Entrada e Saída (*I/O Statistics*), trocas de contexto (*Context Switches*), número total de threads (*Number of*

http://dx.doi.org/10.17058/tecnolog.v20i1.6569
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

*Threads*) e informações gerais de processos (*Process Information*). Já na Figura 3b é possível ver algumas opções para monitorar os discos existentes, como monitoramento d os discos (*Monitor All Disk Devices*), monitoramento total dos discos (*Full monitoring*), realizando a captura de dados de escrita/leitura. Também permite que sejam capturadas apenas informações de leitura (*Monitor Only Reads*) ou escrita (*Monitor Only Writes*).

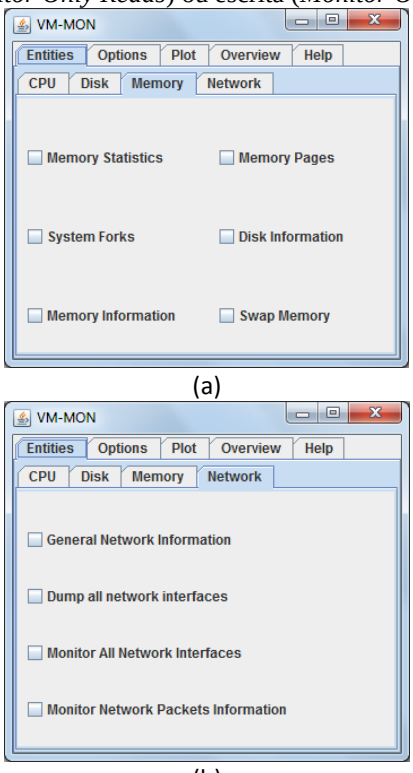


Figura 4 – Monitoramentos da Memória em (a) e monitoramentos da rede em (b).

A Figura 4 continua mostrando a aba *Entities* e em (a) as opções possíveis de Memória (*Memory*) e Rede (Network). Em memória, é possível capturar resultados para estatísticas de memória (*Memory Statistics*), informações sobre páginas (*Memory Pages*), chamadas de *forks* de *kernel* por processos (*System Forks*), algumas informações de disco (*Disk Information*) e memória (*Memory Information*) e por fim, informações de swap (*Swap Memory*). A Figura 4b mostra os monitoramentos possíveis para redes, tais como informações gerais das interfaces (*General Network Information*), captura de dados de passagem pelas interfaces (*Dump all network interfaces*), monitoramento de todas as interfaces (*Monitor all network interfaces*) e monitoramento de pacotes (*Monitor Network Packets Information*).

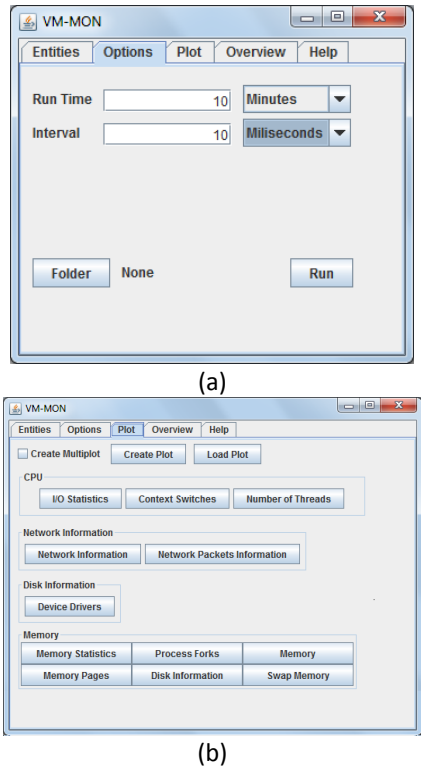


Figura 5 – Detalhamento das opções na aba '*Options*, mostrando as configurações possíveis da ferramenta em (a) e as opções de gráficos em (b).

A Figura 5 mostra as últimas abas da VM-MON, aba *Options* e aba *Plot* (as abas *Overview* e *Help* são utilizadas depois que o usuário escolheu o que deseja monitorar e correspondem a resumos específicos dos comandos bem como um manual de ajuda). A Figura 5a mostra as opções de execução do script de monitoramento, onde se configura o tempo de monitoramento (*Run Time*), podendo escolher entre opções para Segundos, Minutos e Horas, bem como o intervalo de monitoramento (*Interval*) e a pasta que se deseja salvar os arquivos brutos. É nesta aba que o usuário inicia a execução dos monitoramentos escolhidos no botão executar (*Run*). A Figura 5b mostra as opções de gráficos permitidas pelo sistema conforme as coletas efetuadas pelo usuário. É nesta aba que a VM-MON executa o script Perl que extrai os dados e constrói arquivos que serão utilizados por chamadas da ferramenta de criação de gráficos gnuplot.

### *3.5. Discussão*

Ferramentas de monitoramento são vitais para entender a execução de sistemas e prever gargalos e problemas de desempenho. Atualmente, existem diversas maneiras de se monitorar sistemas, entretanto, o problema é que diferentes plataformas exigem diferentes ferramentas auxiliares de monitoração. Neste sentido, a ferramenta descrita neste artigo tenta mitigar este problema oferecendo uma implementação multiplataforma que integra conceitos de monitoramento de

http://dx.doi.org/10.17058/tecnolog.v20i1.6569
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

diferentes SOs. Por si só, esta se constitui em uma inovação importante quanto à maneira de se avaliar desempenho de máquinas físicas e virtuais. Cabe ressaltar que quando monitora-se o desempenho de máquinas virtualizadas na verdade o interesse principal é às questões de divisão dos recursos que são realizadas pelo hipervisor utilizado, ou seja, qual é de fato, o desempenho deste em relação ao desempenho da máquina física.

Além disso, esta ferramenta, se utilizada em modo *stand-alone* em uma determinada arquitetura, funciona como uma interface gráfica de monitoramento, ou seja, pode ser usada para monitorar sistemas de máquinas físicas, de máquinas virtualizadas, e na conjunção destas duas. Da maneira que a ferramenta foi implementada, ela permite a rápida programação e adição de novas ferramentas e parâmetros, servindo como uma alternativa válida para agregar ferramentas auxiliares presentes em diferentes SOs.

## 4 Conclusões

O presente trabalho mostrou as principais considerações necessárias para implementar ferramentas de monitoramento e mostrou o detalhamento de uma ferramenta chamada VM-MON que monitora máquinas físicas e virtualizadas a partir de diferentes sistemas operacionais. O trabalho foca os esforços no monitoramento de máquinas GNU/Linux e MS-Windows, entretanto, não descartamos a implementação de diretivas de programação para agregar novos SOs, tais como Unix ou FreeBSD. Um trabalho futuro interessante que se apresenta aqui seria a implementação de gráficos históricos e implementação de disparos automáticos (no GNU/Linux isso seria feito pelo comando `crontab` e no MS-Windows pelo comando `at`), conforme datas e horas escolhidas pelos usuários. Isso tornaria a ferramenta independente da intervenção direta dos usuários, uma vez que permaneceriam monitorando enquanto os sistemas estivessem executando e conforme as opções dos usuários.

Existem diversas outras funcionalidades que podem ser adicionadas a esta ferramenta, por exemplo, seria possível abrir e fechar diferentes sessões de análise e monitoramento, gerando de forma automática relatórios em formato *Portable Document Format* (PDF) para análise pelos usuários. Também, a ferramenta poderia ter uma versão sem interface gráfica, para sistemas virtualizados que não possuam uma biblioteca visual pré-instalada. Como trabalho futuro, vamos olhar as demais ferramentas do pacote *sysstat* (por exemplo, sar, mpstat, pidstat, etc) do GNU/Linux e estender os contadores de desempenho do MS-Windows para trabalhar com outros monitoramentos interessantes. Seria interessante também executar a ferramenta em um outro hipervisor e comparar seu comportamento com o VirtualBox para avaliar a perda ou não de desempenho conforme as diferentes implementações.

Em termos de aplicação a sistemas e processos industriais, o presente trabalho ramifica algumas questões relevantes, tais como: i) pode ser aplicado em diferentes organizações de TI e indústrias com o objetivo de monitorar diferentes máquinas virtuais de forma ampla; ii) permite testar sistemas desenvolvidos em diferentes plataformas objetivando a mensuração e avaliação de desempenho através da coleta de índices que devem respeitar requisitos de desempenho de sistemas; iii) permite a personalização da ferramenta em diferentes contextos industriais permitindo observar e inspecionar índices específicos conforme o problema sendo analisado. A presente solução destina-se principalmente a sistemas computacionais onde é crucial que sejam calculadas a vazão, população, utilização e tempo de resposta, auxiliando tomadores de decisão a escolher melhores configurações de *hardware* e de *software*.

O trabalho aqui demonstrado é relevante no âmbito da análise de sistemas e monitoramento de recursos de máquinas físicas e virtuais e os trabalhos relacionados evidenciam estas preocupações da comunidade científica em propor e implementar ferramentas leves (*lightweight*), fáceis de usar e escaláveis para utilização em tecnologias de Computação na Nuvem.

## Agradecimentos

O trabalho foi inserido no contexto do projeto *"Exploração de técnicas de avaliação de desempenho em ambientes virtualizados"*, Fluxo 139071, com apoio institucional da UNISC.

## SOFTWARE IMPLEMENTATION CONSIDERATIONS OF MULTIPLATFORM TOOLS APPLIED TO THE MONITORING OF VIRTUALIZED SYSTEMS

**ABSTRACT**: Virtualization through the use of virtual machines over physical machines for systems execution in different domains has been a commonly used approached for several application contexts. The possibility to abstract entire platforms, infrastructures, or software as a service has become a valid technique for execution using Cloud Computing where virtualization is one of the most important technologies to enable it. The utilization of virtualized environments is crucial, however, in several cases, those technologies are chosen with total disregard as to performance considerations or other nonfunctional attributes such as Quality-of-Service (QoS), resilience, reliability, fault tolerance and scalability, to name a few. The objective of the present work focus on the implementation considerations needed to build a complete monitoring tool to inspect virtualized systems. The idea is to estimate better software and hardware configurations to virtualized platforms without performance degradations.

**Keywords**: virtualization, software implementation, monitoring

http://dx.doi.org/10.17058/tecnolog.v20i1.6569



## Referências

[1] WAINER, G. A.: Discrete-event modeling and simulation: a practitioner's approach. CRC Press, 2010.

[2] AVIZIENIS, J. C., RANDELL B., LANDWEHR C.: Basic concepts and taxonomy of dependable and secure computing. In: IEEE transactions on dependable and secure computing, pp. 11-33, 2004.

[3] SOMMERVILLE, I.: Software Engineering. Addison-Wesley, Pearson Education, 2011.

[4] STEWART, W. J.: Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling. Princeton University Press, 2009.

[5] MENASCE, D. A., DOWDY, L. W., ALMEIDA, V. A. F.: Performance by design: computer capacity planning. Prentice Hall PTR Upper Saddle River, NJ, USA, 2004.

[6] CZEKSTER, R. M.: Solução numérica de descritores Markovianos a partir de reestruturações de termos tensoriais. Tese de doutorado, Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS), Programa de Pós-graduação em Ciência da Computação, pp. 1-195, 2010.

[7] JAIN, R.: The Art of Computer Systems Performance Analysis – Techniques for Experimental Design, Measurement, Simulation, and Modeling. Digital Equipment Corporation - Littleton, Massachusetts. John Wiley & Sons, Inc. 1991.

[8] TRIVEDI, K. S., SAHNER, R. A., PULIAFITO, A.: Performance and Reliability Analysis of Computer Systems. Kluwer Academic Publishers, 1996.

[9] TRIVEDI, K. S.: Probability and Statistics with Reliability, Queuing and Computer Science Applications. John Wiley & Sons, Inc., 2002.

[10] STEWART, W. J.: Introduction to numerical solutions of Markov chains. Princeton University Press, 1994.

[11] AJMONE-MARSAN, M., CONTE, G., BALBO, G.: A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems. In: ACM Transactions on Computer Systems, no. 2, pp. 93-122, 1984.

[12] DONATELLI, S.: Superposed stochastic automata: a class of stochastic Petri nets with parallel solution and distributed state space. Performance Evaluation, vol. 18, pp. 21-36, 1993.

[13] HILLSTON, J.: A compositional approach to performance modeling. Cambridge University Press, New York, USA, 1996.

[14] BARHAM, P., DRAGOVIC, B., FRASER, K., HAND, S., HARRIS, T., HO, A., NEUGEBAUER, R., PRATT, I., WARFIELD, A.: Xen and the art of virtualization. In: ACM SIGOPS Operating Systems Review (37), no. 5, pp. 164-177, 2003.

[15] CHOWDHURY, N. M., BOUTABA, R.: A survey of network virtualization. In: Computer Networks, vol. 54, no. 5, pp. 862-876, Elsevier, 2010.

[16] BUYYA, R., BROBERG, J., GOSCINSKI, A. M.: Cloud computing: Principles and paradigms. vol. 87. Wiley, 2010.

[17] MAURO, D. R., SCHMIDT, K. J.: Essential SNMP, Second Edition. O'Reilly Media, Inc., 2005.

[18] ACETO, G., BOTTA, A., de DONATO, W., PESCAPÈ, A.: Cloud monitoring: A survey. In: Computer Networks, Volume 57, Issue 9, pp. 2093-2115, 2013.

[19] FATEMA, K., EMEAKAROHA, V. C., HEALY, P. D., MORRISON, J. P., LYNN, T.: A survey of Cloud monitoring tools: Taxonomy, capabilities and objectives. In: Journal of Parallel and Distributed Computing, vol. 74, no. 10, pp. 2918-2933, 2014.

[20] WARD, J. S., BARKER, A.: Observing the clouds: a survey and taxonomy of cloud monitoring. In: Journal of Cloud Computing 3, no. 1, pp. 1-30, 2014.

[21] MATHEW, J., MCCRAW, H., MOORE, S., MUCCI, P. J., NELSON, J., TERPSTRA, D., WEAVER, V. M., MOHAN, T.: PAPI-V: Performance Monitoring for Virtual Machines. In: International Conference on Parallel Processing Workshops (ICPP'2012), pp. 194-199. 2012.

[22] MONTES, J., SÁNCHEZ, A., MEMISHI, B., PÉREZ, M. S., ANTONIU, G.: GMonE: A complete approach to cloud monitoring. In: Future Generation Computer Systems, vol. 29, no. 8, pp. 2026-2040, 2013.

[23] POVEDANO-MOLINA, J., LOPEZ-VEGA, J. M., LOPEZ-SOLER, J. M., CORRADI, A., FOSCHINI, L.: DARGOS: A highly adaptable and scalable monitoring architecture for multi-tenant Clouds. In: Future Generation Computer Systems, vol. 29, no. 8, pp. 2041-2056, 2013.

[24] SANTOS, M., FERNANDES, C., BENEVENUTO, F., ALMEIDA, V., ALMEIDA, J. VEPMon: Uma Ferramenta de Monitoração de Desempenho para Ambientes Virtuais. In: Simpósio Brasileiro de Redes de Computadores (SBRC) – Salão de Ferramentas, 2008.

[25] DE CARVALHO, M. B., ESTEVES, R. P., RODRIGUES, G. da C., MARQUEZAN, C. C., GRANVILLE, L. Z., TAROUCO, L. M. R. Efficient configuration of monitoring slices for cloud platform administrators. In: 2014 IEEE Symposium on Computers and Communication (ISCC), pp. 1-7, 2014.

http://dx.doi.org/10.17058/tecnolog.v20i1.6569



[26] CALERO, J. M. A., AGUADO, J. G.: Comparative analysis of architectures for monitoring cloud computing infrastructures. In: Future Generation Computer Systems, vol. 47, pp. 16-30, 2015.

[27] DHINGRA, M., LAKSHMI, J., NANDY, S. K.: Resource usage monitoring in clouds. In: Proceedings of the 2012 ACM/IEEE – 13th International Conference on Grid Computing, pp. 184-191. IEEE Computer Society, 2012.

[28] MA, K., SUN, R., ABRAHAM, A. Toward a lightweight framework for monitoring public clouds. In: 4th International Conference on Computational Aspects of Social Networks (CASoN), pp. 361-365. 2012.