JOÃO ANTONIO MAGRI RODRIGUES

Simulação de PaaS no iSPD

Monografia apresentada ao Departamento de Ciências de Computação e Estatística do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", como parte dos requisitos necessários para aprovação na disciplina Projeto Final.

São José do Rio Preto
2015

JOÃO ANTONIO MAGRI RODRIGUES

Simulação de PaaS no iSPD

Monografia apresentada ao Departamento de Ciências de Computação e Estatística do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", como parte dos requisitos necessários para aprovação na disciplina Projeto Final.

Orientador:
Prof. Dr. Aleardo Manacero Junior

São José do Rio Preto
2015

JOÃO ANTONIO MAGRI RODRIGUES

Simulação de PaaS no iSPD

Monografia apresentada ao Departamento de Ciências de Computação e Estatística do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", como parte dos requisitos necessários para aprovação na disciplina Projeto Final.

Prof. Dr. Aleardo Manacero Junior	João Antonio Magri Rodrigues

Banca Examinadora:
Profª. Drª. Renata Spolon Lobato
Prof. Dr. Carlos Roberto Valencio

São José do Rio Preto
2015

Ao meu avô eterno Antonio Magri

AGRADECIMENTOS

Este trabalho marca em minha vida o fim de mais um ciclo, e se cheguei até aqui, não faltou apoio incondicional de pessoas que merecem ser lembradas para sempre nesse texto.

Agradeço em primeiro lugar aos meus pais Edgard Bellini e Zezé, minha irmã Ana Carolina.

Às minhas avós Angélica e Aparecida. Ao meu grande avô, eterno Antonio Magri.

À minha namorada Alana Bugati, pelo carinho e empatia. Às demais pessoas da minha família que sempre estiveram do meu lado: Lucianna, Camila, Daniela, Claudete, Antonio Bellini, José Carlos (Tenente), Victor, Wesley (Tikin) e Rogério.

Aos meus amigos Guilherme Rodrigues, Guilherme (Bazuca), Guilherme (Gaúcho), Victor Hugo, Artur, Nilson, Rafael (Gaúcho), André, Felipe (Baby), Carlos Simão, Fernando (Marrone), Danilo (Fofão), João Marcos, Vinicius (Sagat), Gustavo (Bomba), Guilherme (Wiki).

Ao meu professor de Jiu-Jitsu Thyago Fernandes, por sempre me inspirar a ser melhor.

Ao pessoal do Grupo de Sistemas Paralelos e Distribuídos (GSPD), em especial ao Diogo Tavares, Arthur (Fatal), Denison pela ajuda e companheirismo, aos nossos orientadores, os professores Aleardo Manacero e Renata Spolon, pela paciência e orientação durante o tempo que estive em iniciação científica.

Finalmente, agradeço a UNESP pelo apoio através das bolsas PIBIC/Reitoria.

“Ser tão forte que nada possa perturbar a paz da tua mente.”
- Carlos Gracie

RESUMO

O iSPD (*iconic Simulator of Parallel and Distributed Systems*) é um simulador de grades computacionais, que diferente dos demais simuladores, dispõe ao usuário uma interface icônica para a criação de modelos de simulação de uma forma ágil e intuitiva. O objetivo deste trabalho é implementar no iSPD um módulo capaz de simular um serviço de PaaS (*Platform as a Service*), que consiste em uma classe de serviço oferecida através de um sistema de computação em nuvem.

ABSTRACT

iSPD (iconic Simulator of Parallel and Distributed System) is a grid computing simulator that, unlike others simulators, provides a graphic interface to its user in order to create simulation models in a quick and intuitive way. The objective of this work is to implement in iSPD a module able to simulate the class of service PaaS (Platform as a Service), which refers to a service provided through a cloud computing system.

ÍNDICE

Lista de Figuras

Lista de Tabelas

1 INTRODUÇÃO

Computação em nuvem (do inglês *Cloud Computing*) é o nome que se dá a um tipo sistema de computação composto por uma coleção de máquinas virtuais interconectadas e provisionadas dinamicamente, as quais tem sua disponibilidade baseada em um contrato de nível de serviço [Buyya et al, 2011].

Do ponto de vista do cliente, a computação em nuvem tem como proposta um novo modelo de estruturação de TI, transferindo sua complexidade, desde servidores a pacotes de software, para a nuvem. Ou seja, o proprietário lança mão da aquisição de máquinas físicas e pacotes de software para alugar esses recursos de um provedor de computação em nuvem, necessitando apenas de um navegador *web* (ou até mesmo aplicações específicas fornecidas pelo provedor) para utilizar e gerenciar seus recursos.

Devido a sua diversidade de soluções, a computação em nuvem é capaz de atender a diversos nichos de mercado, desde a locação de servidores a nível empresarial até *backup* de arquivos pessoais. Esses fatores aliados à eficiência e custo-benefício têm popularizado cada vez mais os serviços de computação em nuvem.

Em função da complexidade desse tipo de sistema, é imprescindível que haja ferramentas capazes de realizar uma análise de seu desempenho. Para tanto, existem simuladores capazes de avaliar o desempenho de sistemas computacionais a partir de modelos que representam uma estrutura de um sistema. Para computação em nuvem já existem ferramentas poderosas para a realização dessa tarefa, tais como CloudSim [Buyya; Ranjan; Calheiros, 2009], iCanCloud [Castane; Nunez; Carretero, 2012], GreenCloud [Kliazovich, 2010].

O iSPD (*iconic Simulator of Parallel and Distributed Systems*) [Manacero et al, 2012] é um simulador de grades computacionais desenvolvido pelo GSPD (Grupo de Sistemas Paralelos e Distribuídos) da Unesp, capaz de simular serviços de computação em nuvem. A versão do simulador apresentada neste trabalho contempla a implementação de novos módulos a fim de promover a simulação da classe de serviço PaaS.

1.1 Motivação

O projeto de sistemas de computação em nuvem não é uma tarefa simples. Além de demandar conhecimentos nas tecnologias relacionadas da área, é necessário um estudo estratégico a fim de se aproveitar o máximo da qualidade dos recursos disponíveis para a demanda esperada.

No entanto, a execução de *benchmarks* nem sempre é a solução mais adequada para esse problema. *Benchmarks* necessitam de um sistema real para serem executados e por vezes vem a se tornar um processo caro e demorado [Silva et al, 2015]. Para sanar esse problema existem os simuladores, que são ferramentas capazes de promover uma avaliação de antemão de forma barata e precisa a cerca de um modelo que apenas represente um sistema, o qual pode existir ou não.

O iSPD é um simulador que tem como diferencial facilitar e agilizar a criação de modelos de simulação por meio de uma interface gráfica para a interação com seu usuário.

1.2 Objetivo

O Objetivo deste trabalho é adicionar ao iSPD novas funcionalidades para a caracterização de um serviço de PaaS no processo de simulação.

O iSPD já possui a simulação da camada de virtualização de um sistema de computação em nuvem, para tanto, as novas funcionalidades contam com novas políticas de alocação e provisionamento dinâmico para melhor caracterizar um serviço de PaaS.

1.3 Organização do texto

No primeiro capítulo é feita uma introdução ao tema, apresentando superficialmente alguns conceitos a cerca de computação em nuvem, bem como a importância do processo de simulação.

O segundo capítulo trata da fundamentação teórica relativa ao tema na qual se baseou a implementação do projeto.

Em seguida, apresenta-se de fato a implementação dos novos módulos no simulador iSPD.

Após apresentar a implementação, há um capítulo dedicado a testes realizados nos simuladores iSPD e CloudSim, comparando os resultados gerados por ambas as ferramentas.

E finalmente, o último capítulo dedicado às conclusões e considerações finais.

2 REVISÃO BIBLIOGRÁFICA

Em um processo de simulação de um sistema, podemos ter infinitos pontos de vistas para gerar uma análise. É de suma importância conhecer as características chaves do serviço alvo, tal como o comportamento do sistema se ajusta ao contrato de nível de serviço, para poder gerar métricas de um lugar comum.

Neste capítulo apresentar-se-á estudos a respeito de computação em nuvem, mais especificamente voltados para o serviço de PaaS. Há também seções dedicadas a apresentação do simulador iSPD e de outras ferramentas relacionadas.

2.1 Conceitos sobre computação em nuvem

Computação em nuvem é um conceito relativamente recente, que tem ganhado cada vez mais espaço no mercado de TI. [Buyya et al] define computação em nuvem como uma coleção de máquinas virtuais interconectadas e provisionadas dinamicamente, disponíveis a partir de um contrato de nível de serviço.

Portanto, a virtualização é um conceito muito importante dentro do contexto de computação em nuvem e será tratada a seguir, juntamente com as características essenciais e as classes de serviços que compõem um sistema de computação em nuvem.

2.1.1 Características Essenciais

De acordo com [Mell; Grance, 2011] em um documento publicado pela NIST (*National Institute of Standards and Technology*), um sistema de computação em nuvem é composto por cinco características essenciais. São elas:

- **Autoatendimento:** O cliente é capaz de contratar, gerenciar e pagar pelo serviço sem a necessidade de interação humana por parte do provedor.
- **Acesso remoto:** O cliente pode acessar os recursos contratados através de uma rede de computadores através de diversos dispositivos de computação, como *smartphones*, *laptops*, *tablets.*
- **Agrupamento de recursos:** Os recursos computacionais são agrupados para atender diversos usuários, usando técnicas de provisionamento dinâmico para ajustar melhor a dimensão de recursos à demanda de clientes. A disposição dos recursos na rede é totalmente transparente ao cliente e não gera conflitos operacionais.
- **Elasticidade ágil:** A capacidade de recursos pode ser alterada de forma rápida para melhor ajustar a demanda. Há possibilidade tanto de aumentar como diminuir os recursos para melhorar o desempenho do sistema.
- **Medição de serviço:** O provedor utiliza métricas para redimensionar os recursos e cobrar pela utilização de um determinado serviço, a fim de justificar sua cobrança.

2.1.2 Classes de serviço

Um serviço de computação em nuvem agrega três classes essenciais. São elas:

- ***Software as a Service*** **(SaaS):** Diferente do modelo tradicional de distribuição de *softwares*, em que o produto adquirido é instalado no computador do cliente (*Software as a Product*), nesta classe de serviço o *software* reside no centro de serviço do seu provedor e é acessado remotamente por seu cliente, normalmente pela Internet [Rittinghouse et al, 2009]. Dentro do contexto de computação em nuvem, essa classe de serviço refere-se a aplicações web direcionadas a usuários comuns, tal como editores de textos, MUAs (*Mail User Agent*) e até mesmo *front-end* para a configuração de outras classes de serviço.

- ***Platform as a Service*** **(PaaS):** Nesta classe de serviço o provedor disponibiliza ao cliente um ambiente para desenvolvimento e implantação de aplicações com um alto nível de abstração, de forma que o cliente não necessite conhecer detalhes inerentes a plataforma que esta utilizando [Buyya et al, 2011]. Assim, fica a cargo do provedor o gerenciamento e configuração dos recursos. Uma das características chave de um serviço de PaaS é a disponibilidade de uma aplicação *web* como *front-end* para o apoio necessário ao ciclo de vida de desenvolvimento de *softwares* [Rittinghouse et al, 2009].
- ***Infrastructure as a Service*** **(IaaS):** Esta é a classe de serviço em que o provedor oferece o menor nível de abstração ao cliente. Diferente do serviço de PaaS, no IaaS a responsabilidade de configuração e gerenciamento de recursos é delegada ao cliente. Por outro lado, o cliente tem maiores privilégios sobre o servidor, tais como a instalação arbitrária de pacotes de *softwares,* anexação de discos virtuais, implementação de políticas de *firewall* [Buyya et al, 2011].

2.1.3 Camada de virtualização

Como já discutido anteriormente, um sistema de computação em nuvem consiste em uma coleção de máquinas virtuais implantada sobre um sistema distribuído de máquinas físicas. Essa coleção de máquinas virtuais, quando interconectadas recebem o nome de *cluster* virtual, pois possuem as mesmas propriedades de um *cluster* físico, tal como processamento paralelo e balanceamento de carga.

O dispositivo responsável pela virtualização e gerenciamento de recursos é conhecido como camada de virtualização. Esse dispositivo é responsável por alocar máquinas virtuais e escalonar tarefas para as mesmas.

O *software* que implementa essa funcionalidade é chamado VMM (*Virtual Machine Manager*) ou *hypervisor*. [Buyya et al, 2011]. São exemplos desses *softwares* Xen [Xen Project, 2015] e VMWare [VMware, 2015].

Um sistema de computação em nuvem dispõe de muitas máquinas físicas aptas a hospedar uma máquina virtual. A camada de virtualização é também responsável por selecionar uma máquina física hospedeira para uma máquina virtual. Para tanto, existem algoritmos para a realização dessa tarefa. Alguns desses algoritmos são:

- ***Round-Robin:*** Neste algoritmo as máquinas virtuais são instanciadas uniformemente entre os recursos físicos. Para isso, os recursos são organizados em um lista circular e selecionados um a um para receber uma máquina virtual.
- ***First-Fit:*** Neste algoritmo, as máquinas virtuais são instanciadas no primeiro recurso físico apto a hospeda-la encontrado pelo VMM.
- ***First-Fit Decreasing:*** Este método é similar ao *First-Fit* previamente descrito, no entanto, as solicitações de máquinas virtuais são organizadas em uma lista ordenada de ordem decrescente em relação à quantidade de recursos solicitada por cada máquina virtual para que as solicitações sejam atendidas de forma ordenada.
- ***Volume:*** Este é outro algoritmo que implementa o conceito de *First-Fit*, no entanto, os recursos físicos são organizados através de uma lista decrescente de acordo com o produto das dimensões que definem capacidade de armazenamento e poder computacional. O encaminhamento de solicitação de máquinas virtuais aos recursos obedecerá a ordem dessa lista.

Portanto, a camada de virtualização é um dispositivo crítico em um sistema de computação em nuvem e seu projeto pode influenciar diretamente seu desempenho.

2.2 Provisionamento Dinâmico

Um sistema de computação em nuvem esta sujeito a oscilações de demanda, tal como a chegada de novos clientes, aumento não previsto do trafego de aplicações hospedadas, entre outros fatores que podem causar sobrecarga em seus recursos. O sistema deve ter a capacidade de provisionar novos recursos dinamicamente e desalocar recursos ociosos de forma ágil, transparente e automática a fim de conservar a qualidade de serviço.

A implementação de funcionalidades que propiciam a elasticidade de recursos é conhecido como *Autoscaling.* Junto a isso, o sistema também deve proporcionar um monitoramento e balanceamento de carga em seus recursos.

2.2.1 Monitoramento de recursos

Para determinar se uma aplicação necessita de mais ou menos recursos o sistema de computação em nuvem que a hospeda realiza o monitoramento dos seus recursos e os escala de acordo com as suas próprias políticas.

Uma política de escalamento de recursos pode agregar diversos critérios, tais como uso de memória, tamanho de filas, carga computacional e até mesmo critérios probabilísticos, como eventos sazonais [Lorido-Botrán et al, 2012].

Quando há a adoção de critérios baseados em métricas de computação (tamanho de filas, uso de memória, etc) define-se um valor limiar para aumentar os recursos e outro para diminuir. Para ilustrar com um exemplo de algoritmo, definiremos esses valores como *AltoLimiar* e *BaixoLimiar* respectivamente, e *medição* como o valor de uma métrica de computação escolhida. Então, para o aumento de recursos tem-se o algoritmo (1) e para a redução tem-se o algoritmo (2).

Se *medição* > *AltoLimiar* então aumentar recursos (1)

Se *medição* < *BaixoLimiar* então diminuir recursos (2)

Há possibilidade também de agregar outros critérios, tais como intervalo mínimo entre medições, número máximo de instancias, etc.

Alguns provedores possuem políticas próprias para a realização de *Autoscaling*, outros possibilitam o usuário personalizar políticas para realizar essa tarefa.

2.2.2 Autoscaling

A funcionalidade responsável pela elasticidade automática em sistemas de computação em nuvem é conhecida por *Autoscaling*.

As técnicas de *Autoscaling* operam diretamente sobre os recursos virtualizados, portanto, essas técnicas são integradas à camada de virtualização do sistema.

Quanto à natureza dessas técnicas, podemos classifica-las em dois tipos [Lorido-Botrán et al, 2012]:

- **Vertical:** Quando há alteração da capacidade computacional de recursos em particular. Por exemplo, aumento ou diminuição de memória *RAM* e alocação ou desalocação de núcleos virtuais para uma ou mais máquinas virtuais.
- **Horizontal:** Quando há alteração do número de máquinas virtuais de um cluster virtual específico. Por exemplo, quando necessário, aloca-se mais uma máquina virtuais para um cluster, caso contrário, se o sistema detecta que há um número demasiado de máquinas virtuais, então ocorre uma desalocação das mesmas.

Por questões de viabilidade, a técnica mais empregada é a horizontal. Técnicas de *Autoscaling* vertical estão restritas a propriedades do sistema operacional hospedado pela máquina virtual alvo, além de serem mais suscetíveis a erros em relação às técnicas de *Autoscaling* horizontal.

2.2.3 Balanceamento de cargas

Como já mencionado, os recursos virtuais de um sistema de computação em nuvem possuem funcionamento similar a um *cluster* de máquinas físicas, sendo assim nomeado *cluster* virtual.

Dependendo do serviço contratado, um *cluster* virtual pode ser definido para uma aplicação, como ocorre em serviços de SaaS e PaaS ou para fins arbitrários definidos pelo cliente, como em serviços de IaaS.

Para balancear cargas, cada *cluster* virtual possui um *front-end*, que é um dispositivo responsável por determinar para qual nó rotear uma tarefa [Hung et al, 2012].

Os critérios de balanceamento de cargas dependem diretamente de finalidade para a qual o *cluster* virtual esta servindo. Por exemplo, para servidores *web* adotam-se critérios baseados em número de conexões ativas por instância.

Geralmente esses critérios estão relacionados aos critérios usados no monitoramento de recursos para a realização de *Autoscaling*.

2.3 Platform as a Service (PaaS)

Platform as a Service (PaaS) consiste em uma classe de serviço oferecida por provedores de computação em nuvem.

Nesta classe de serviço, o provedor dispõe ao cliente ferramentas para desenvolvimento de aplicações, como ambientes de desenvolvimento integrado para linguagens de programação.

As aplicações desenvolvidas são implantadas no centro de serviço do provedor, que se responsabiliza por todo o gerenciamento de infraestrutura, desde a instalação de plataformas, sistemas operacionais, até o provisionamento dinâmico de recursos.

Por outro lado, o cliente não realiza o gerenciamento da infraestrutura, tal como faz o provedor [Mell; Grance, 2011].

Podemos destacar como exemplo desses serviços o Google App Engine [App Engine, 2015], oferecido pelo Google e o Microsoft Azure [Microsoft Azure, 2015], oferecido pela Microsoft.

2.3.1 Características essenciais

Alguns estudiosos definem PaaS como um tipo especializado de SaaS, por este oferecer um *front-end* baseado em *web* para suporte a todo o ciclo de desenvolvimento de software, proporcionando ao seu usuário um alto nível de abstração [Rittinghouse et al, 2009].

Alguns serviços também possibilitam integrar o processo de desenvolvimento com outros ambientes de desenvolvimento, conferindo maior agilidade e eficiência ao ciclo.

Quanto à implantação de aplicações, são provisionadas dinamicamente máquinas virtuais no centro de serviço do provedor, que é responsável pelo gerenciamento de servidores e garantia de qualidade de serviço.

2.3.2 Oferta e cobrança de serviço

Na contratação de um serviço de PaaS, o cliente escolhe um perfil de máquina virtual, na qual sua aplicação será implantada. Cada perfil possui um valor agregada de acordo com a qualidade de serviço em que nele é oferecida.

Como a qualidade de serviço é responsabilidade do provedor, os recursos computacionais são provisionados dinamicamente, de acordo com critérios próprios do provedor ou definidos pelo cliente.

Para tanto, muitos provedores adotam como a somatória dos produtos entre o custo do perfil escolhido e o tempo que a máquina virtual ficou ativa. Na equação (3) é representado o calculo do custo nesse tipo de cobrança, em que *CustoTotal* é o custo total de uso do serviço, *Cp* é o preço do perfil escolhido e tVM_i o tempo que uma máquina virtual específica ficou ativa.

$$CustoTotal = Cp * \sum_{i=0}^{n} tVMi \quad (3)$$

2.3.3 Aspectos simuláveis

Até o momento foi definido que o serviço de PaaS consiste em ferramentas de apoio ao desenvolvimento de aplicações e a hospedagem das mesmas no centro de serviço do provedor.

A simulação de PaaS consiste em simular a implantação de aplicações, abordando seus aspectos essenciais, como políticas de provisionamento dinâmico, avaliação de qualidade de serviço e cálculo de custos.

A diferença crucial que há entre a simulação de IaaS e PaaS é o escalonamento de tarefas. Em uma simulação de IaaS são definidos *clusters* virtuais por usuários, e as tarefas são escalonadas entre seus nós virtuais. Em PaaS, as aplicações são escalonadas por aplicação, ou seja, cada aplicação terá seu próprio *cluster* virtual que só escalonará tarefas relativas a esta, independente de quem seja o usuário.

2.4 Simuladores de computação em nuvem

A simulação é um procedimento presente em diversas áreas de conhecimento. É um procedimento que busca estimar a eficiência de sistemas de antemão à sua concepção.

Em computação em nuvem não é diferente. O projeto de um sistema pode demandar grandes gastos financeiros. Portanto, a simulação se torna um recurso crítico dentro desse contexto, e especificamente para sistemas de computação, uma solução mais rápida de barata do que a realização de *benchmarks*.

O iSPD a princípio foi concebido como uma ferramenta de simulação de grades computacionais. Em sua última versão, foram implementados novas funcionalidades, capaz de simular um serviço de IaaS. Portanto, o iSPD já possui a implementação da

camada de virtualização, bem como modelos representativos de máquinas virtuais e cargas computacionais.

O simuladores de computação em nuvem mais notórios que há atualmente são CloudSim [Buyya; Ranjan; Calheiros, 2009], iCanCloud [Castane; Nunez; Carretero, 2012], GreenCloud [Kliazovich, 2010]. A seguir, serão apresentados essas ferramentas juntamente com o iSPD.

2.4.1 CloudSim

O *CloudSim* consiste em um conjunto de bibliotecas da linguagem Java desenvolvida pelo grupo CLOUDS (*The Cloud Computing and Distributed Systems*) da Universidade de Melbourne, na Austrália.

As classes presentes nos pacotes de bibliotecas *CloudSim* representam elementos de uma simulação de computação em nuvem, tais como máquinas físicas, máquinas virtuais, cargas computacionais, políticas de escalonamento, etc. Através do conceito de orientação de objeto, é possível criar extensões dessas classes para a personalização desses elementos.

O *CloudSim* também oferece suporte para avaliação do consumo de energia dos sistemas modelados. Seu motor de simulação oferece suporte para interrupção e retomada do processo de simulação.

Portanto, o *CloudSim* é um ferramenta poderosa, no entanto, exige de seu usuário um nível razoável de conhecimento do paradigma de programação orientada a objetos, especificamente em linguagem Java.

Entre os simuladores apresentados, o *CloudSim* foi escolhido para a comparação de resultados apresentados pelo iSPD, que serão apresentados nas sessões de testes e resultados do trabalho.

2.4.2 iCanCloud

O *iCanCloud* é um plataforma para modelagem e simulação de sistemas de computação em nuvem desenvolvida pelo grupo ARCOS (*Grupo de Arquitectura de*

Computadores, Comunicaciones y Sistemas) da universidade Carlos III de Madrid, na Espanha. O simulador trata-se de

O simulador foi desenvolvido em OMNet++ e INET [OMNet++, 2015] e sua utilização demanda o conhecimento do uso desses *frameworks*.

Seu objetivo é realizar avaliações, voltadas para a classe de serviço IaaS, da relação de custo e desempenho de aplicações executadas nos recursos do sistema modelado.

O *iCanCloud* oferece suporte para a implementação de políticas personalizadas e avaliação de consumo de energia dos recursos em um sistema de computação em nuvem.

2.4.3 GreenCloud

O *GreendCloud* é uma ferramenta desenvolvida pela Universidade de Luxemburgo. Trata-se de uma extensão do simulador de redes NS2, possui uma implementação híbrida, em que a maior parte do seu código foi escrita em C++ e o restante em *scripts* TCL (*Tool command language).*

O simulador é completamente voltado para a avaliação de gastos de energia elétrica dos sistemas simulados, atendendo ao conceito de computação verde [Beloglazov; Buyya, 2012], termo que designa o uso sustentável da computação do ponto de vista de consumo de energia.

2.4.4 iSPD

O iSPD (*iconic Simulator of Parallel and Distributed Systems*) é um simulador de grades computacionais e sistemas de computação em nuvem desenvolvido pelo GSPD (Grupo de Sistemas Paralelos e Distruibuídos) da UNESP.

Seu objetivo é democratizar o uso do simulador, oferecendo ao usuário uma interface gráfica para a criação de modelos de simulação de forma rápida e intuitiva. Para tanto, o iSPD possui uma interface gráfica para a interação com seu usuário, onde é possível modelar um centro de serviço juntamente com suas cargas computacionais. Na Figura 2.1 é representada a interface gráfica para a modelagem da infraestrutura física de um sistema de computação em nuvem.

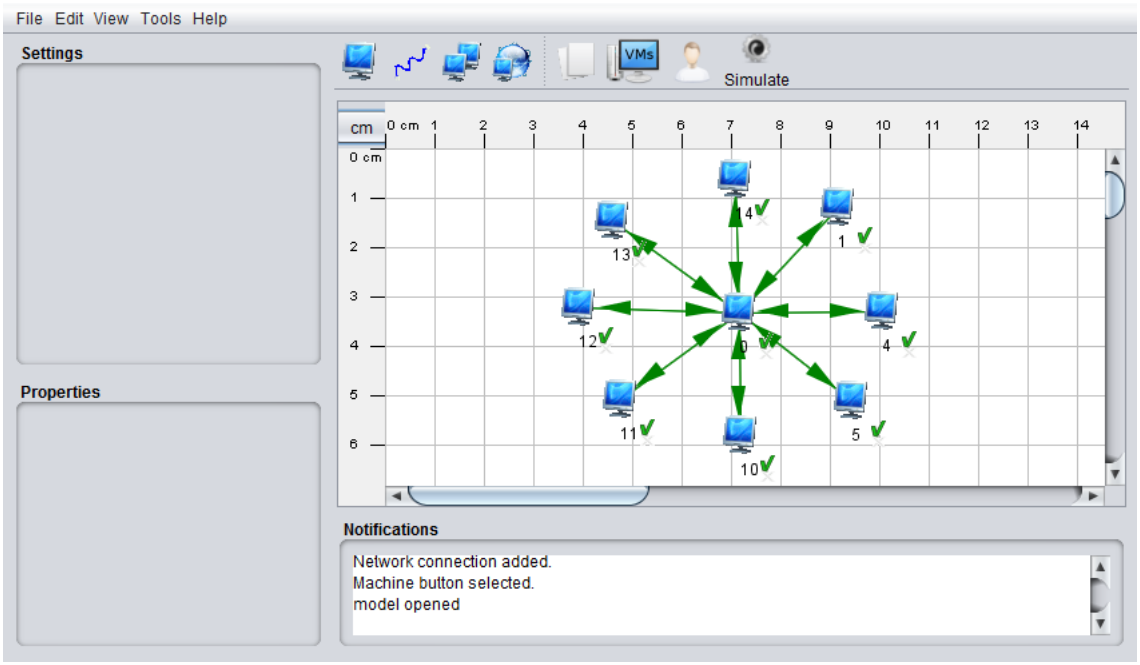

Figura 2.1 : Interface de modelagem de infraestrutura física de um sistema de computação em nuvem.

Os ícones apresentados na parte de cima da Figura 1 representam os componentes da infraestrutura, que são, da esquerda para a direita, computador, *link*, *cluster*, *internet*, tarefas, máquinas virtuais, usuários e por fim, o botão que inicia a simulação do sistema. A área quadriculada abaixo dessa barra de ferramenta representa o espaço disponível para a inserção de elementos físicos.

O funcionamento do iSPD pode ser particionado em cinco módulos essenciais. São esses:

- **Interface icônica:** Interface de interação com o usuário, em que se realiza a inserção dos elementos de simulação.
- **Interpretação de modelos internos:** Módulo responsável por coletar os parâmetros de modelos de simulação criados no iSPD e envia-los a o motor de simulação.
- **Interpretador de modelos externos e exportador de modelos internos:** Neste módulo os modelos de outros simuladores são traduzidos para o iSPD, bem como os modelos do iSPD são traduzidos para outros simuladores. Atualmente o iSPD é capaz de interpretar modelos do GridSim e do SimGrid e traduzir seus modelos para o simulador GridSim.

• **Gerador e Gerenciador de Escalonadores:** Módulo que oferece ao usuário a criação de novas politicas de escalonamento de tarefas.

• **Motor de simulação:** Este é o módulo responsável por realizar a simulação a partir de um modelo criado ou importado pelo usuário.

A interação dos módulos mencionados com o usuário é ilustrada na Figura 2.2.

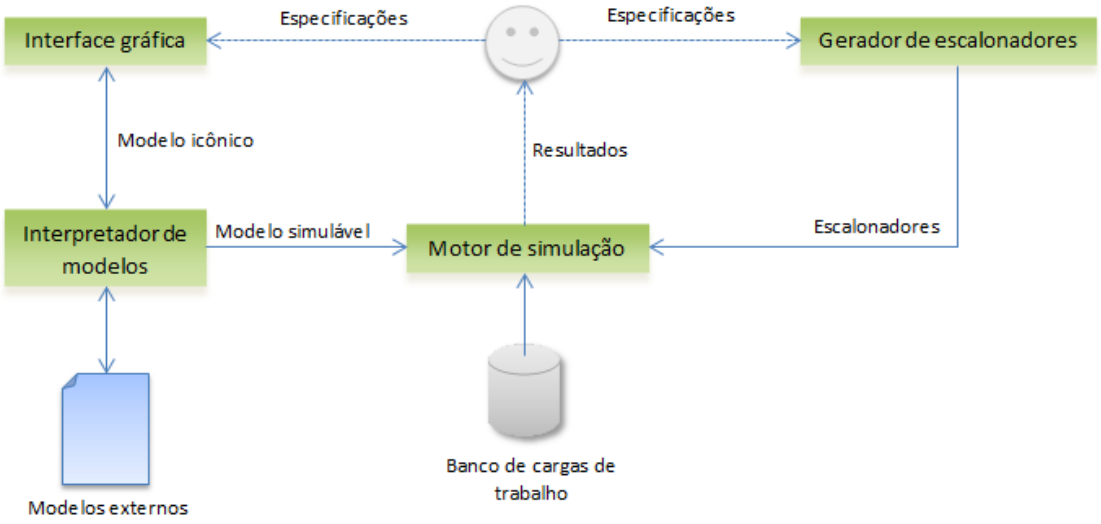

Figura 2.2 : Interação dos módulos do iSPD com o usuário.

Os modelos de simulação do iSPD são armazenados em arquivos formatados em linguagem de marcação XML, onde cada elemento inserido no modelo de simulação é representado por uma *tag*.

Há uma diferença crucial do iSPD para os demais simuladores apresentados. Enquanto outros simuladores consistem em bibliotecas de *frameworks* e linguagens de programação, o iSPD é um software de fato, desenvolvido e compilado em linguagem Java, conferindo portabilidade e facilidade de uso.

3 IMPLEMENTAÇÃO DE NOVAS FUNCIONALIDADES

Neste capítulo apresenta-se o desenvolvimento de novos módulos responsáveis por realizar a simulação do serviço de PaaS no iSPD, com base nos conceitos apresentados no capítulo anterior.

A implementação consiste na adição dos seguintes itens no projeto:

- **Novas interfaces gráficas:** As novas interfaces gráficas são responsáveis por auxiliar o usuário na modelagem de elementos que caracterizem a classe de serviço PaaS.
- **Novas classes:** As novas classes armazenam os dados relativos à modelagem de recursos e aplicações em PaaS.
- **Especificação de escalonadores:** Para caracterizar o serviço de PaaS, as tarefas devem ser escalonadas estritamente por aplicação, e não mais apenas por usuário, como ocorreu na última versão do iSPD.
- **Provisionamento dinâmico:** Esta nova funcionalidade implementa os conceitos de provisionamento dinâmico apresentados no capítulo anterior, juntamente com uma política específica de *autoscaling* e balanceamento de cargas.
- **Novas métricas:** Para a análise do serviço de PaaS, são necessárias novas métricas para gerar uma avaliação a partir de um novo ponto de vista que caracterize um serviço de PaaS.

As sessões a seguir dedicam-se a apresentam o processo de implementação de cada um desses itens.

3.1 Novas interfaces gráficas

Na Figura 2.1 representou-se a interface de modelagem de recursos físico no iSPD. Ao pressionar o botão que representa máquinas virtuais tem-se o acesso à interface de modelagem de serviços em nuvem, representada pela Figura 3.3.

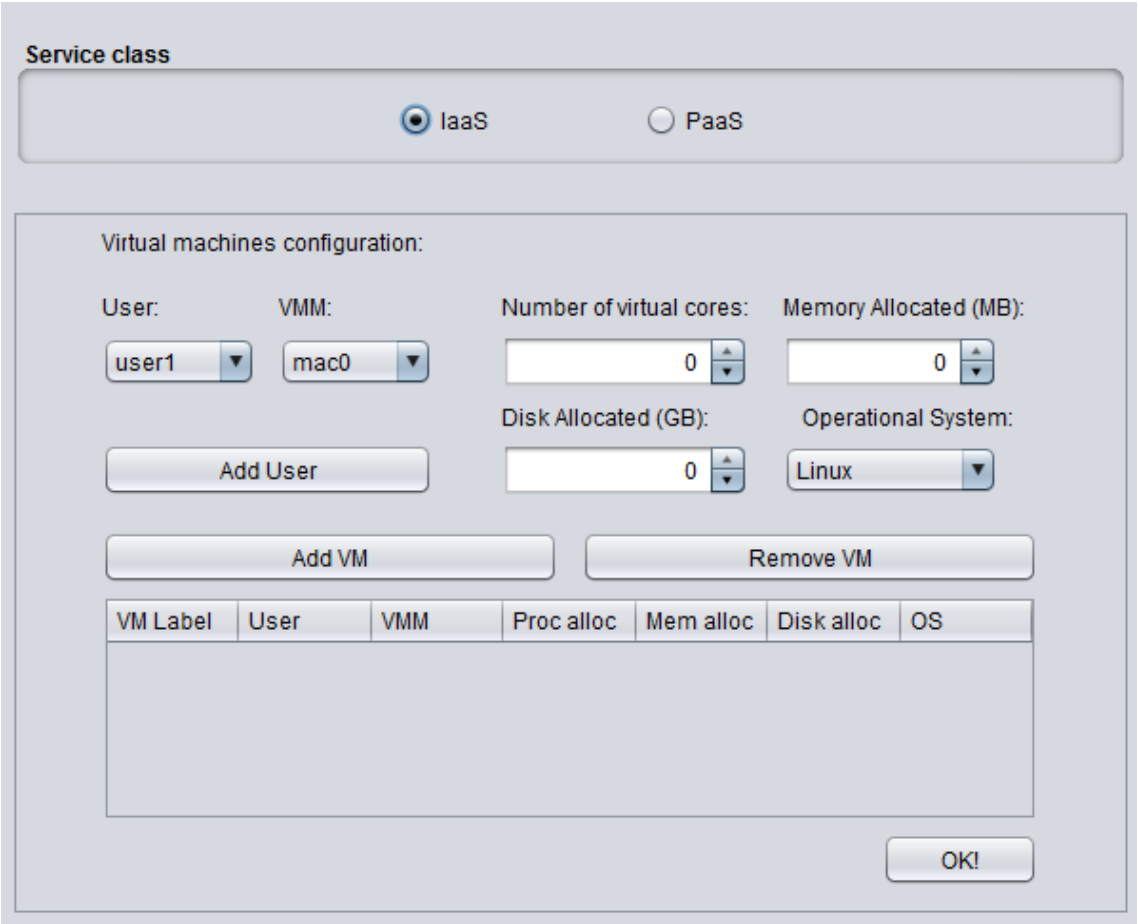

Figura 3.3: Interface de modelagem de serviços em nuvem

Por padrão, a interface que aparece inicialmente é referente a modelagem do serviço IaaS. Para acessar a interface referente ao serviço de PaaS, basta pressionar o botão rotulado PaaS. Ao realizar esse procedimento, obtemos a interface representada na Figura 3.4.

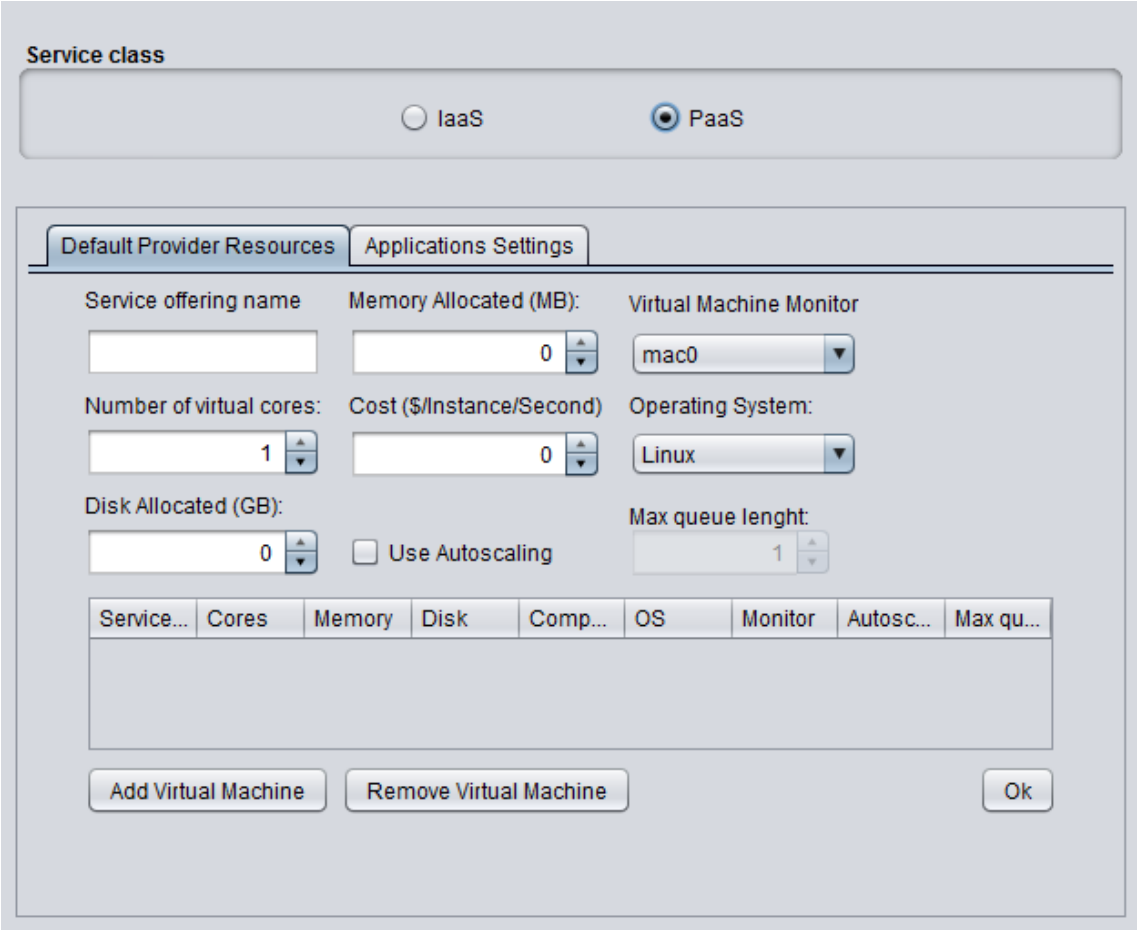

Figura 3.4: Interface de modelagem de serviços em nuvem (PaaS)

O Processo de modelagem da classe de serviço PaaS divide-se em duas etapas, que são separadas em diferentes abas. Essas abas representam:

- ***Default Provider Resources:*** Modelagem perfis de recursos que serão agregados a uma ou mais aplicações.
- ***Applications Settings:*** Modelagem de aplicações que serão executadas no centro de serviço do sistema modelado.

Cada uma dessas interfaces possui muitos componentes. Para facilitar a explanação, serão dedicadas duas subseções, uma para cada interface, explicando com maior nível de detalhamento a função de seus componentes.

3.1.1 Modelagem de perfis de máquinas virtuais

Esta interface é acessada através da aba *Default Provider Resources*. Por padrão, é a primeira a ser exibida ao selecionar modelagem da classe de serviços PaaS, e portanto, esta representada na Figura 3.4.

Seus componentes são representados pelos através dos seguintes rótulos:

- ***Service offering name:*** Uma cadeia de caracteres única que representa cada perfil de máquina virtual modelado.
- ***Memory Allocated (MB):*** Quantidade de memória primária em MB alocada para a o perfil de máquina virtual.
- ***Virtual Machine Monitor:*** Servidor que hospeda o *front-end* do *hypervisor* desse perfil de máquina virtual. Esse servidor é responsável pela instanciação de máquinas virtuais nos seus demais escravos e escalonamento de tarefas para as mesmas.
- ***Number of virtual cores:*** Número de núcles virtuais que terá esse perfil de máquina virtual.
- ***Cost($/Instances/Seconds):*** Custo por segundo que terá cada instancia de máquina virtual pertencente a esse perfil.
- ***Operating System:*** Sistema operacional escolhido para esse perfil de máquina virtual.
- ***Disk Allocated (GB):*** Quantidade de memória secundária em GB alocada para esse perfil de máquina virtual.
- ***Use Autoscaling:*** Caixa de seleção que determina se o perfil modelada usará *autoscaling.*
- ***Max queue lenght****:* Caso o opção de *autoscaling* esteja ativa, o usuário pode determinar um tamanho máximo de fila, que caso a tamanho da fila exceda esse valor, a máquina virtual solicita *autoscaling*.
- ***Add Virtual Machine:*** Adiciona o perfil modelado a lista de serviços oferecidos, exibida através da única tabela que há na interface.
- ***Remove Virtual Machine:*** Caso o usuário opte por excluir um ou mais perfis modelados, basta selecionar o(s) perfil(s) desejado e na lista e pressionar este botão.
- ***Ok:*** Salva as alterações e fecha a janela.

3.1.2 Modelagem de aplicações

Esta interface é acessada através da aba *Applications Settings*. Ela possibilita a modelagem de aplicações que serão simuladas no centro de serviço do modelo criado. O iSPD entende por aplicações um número finito de tarefas, que possuem um carga computacional definida e um destino específico.

Na Figura 3.5 é representada essa interface.

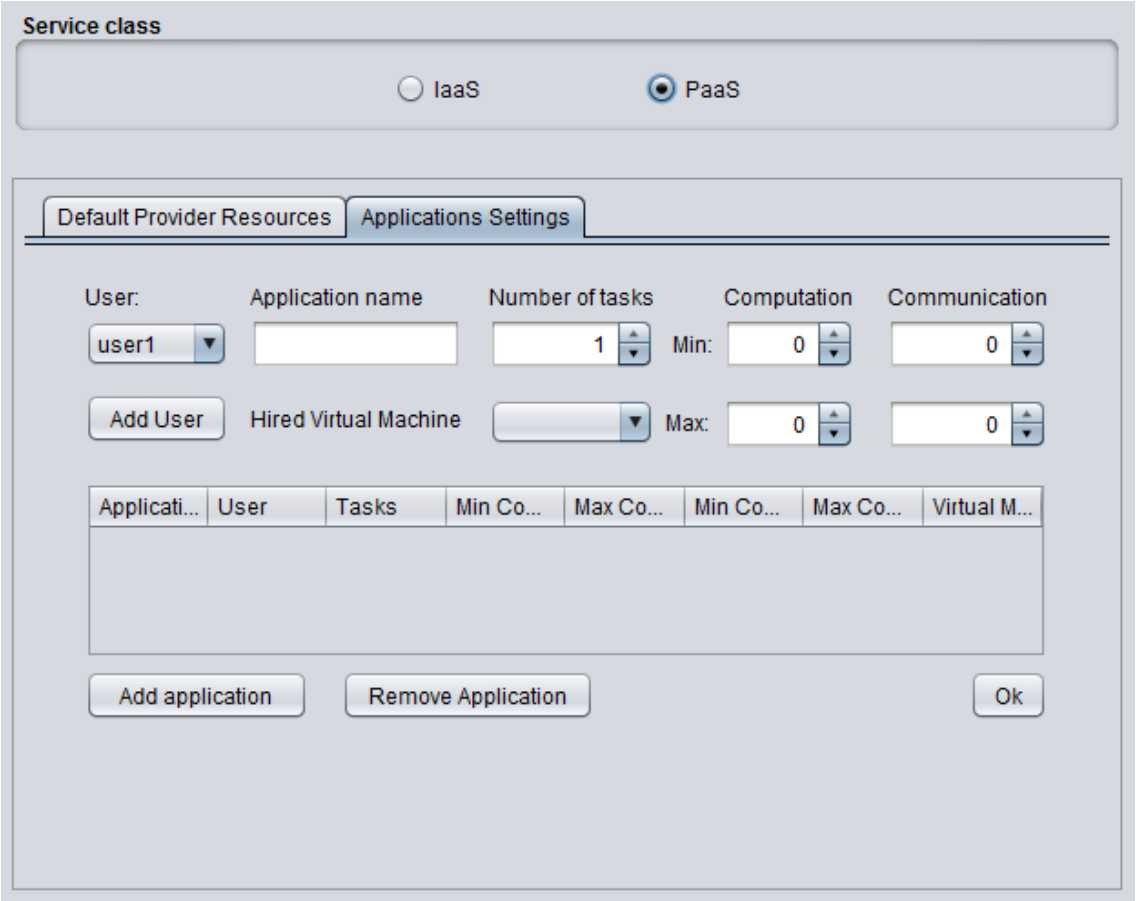

Figura 3.5: Interface de modelagem de aplicações em PaaS.

Os rótulos representam os seguintes componentes dessa interface:

- ***User:*** Usuário a qual pertence à aplicação modelada.
- ***Application Name:*** Cadeia de caracteres única que representa a aplicação modelada.
- ***Add User:*** Adiciona um usuário no modelo de simulação.
- ***Number of tasks:*** Número de tarefas que possui essa aplicação.
- ***Computation:*** Carga computacional de cada tarefa integrante dessa aplicação. A carga é representada em MFlop. Os rótulos *Min* e *Max* representam respectivamente os valores mínimo e máximo que essa grandeza pode assumir.
- ***Communication:*** Tamanho de comunicação de cada tarefa. Esta grandeza é representada em *bytes* e assim como a carga computacional pode assumir um valor contido no intervalo definido por *Min e Max.*

- ***Hired Virtual Machine:*** Perfil de máquina virtual escolhido para executar essa aplicação.
- ***Add Application:*** Adiciona a aplicação modelada ao modelo de simulação. Essa aplicação aparece disponível na única tabela presente na interface.
- ***Remove Application:*** Remove uma aplicação selecionada na tabela de aplicações modeladas.
- ***Ok:*** Salva as alterações e fecha a janela.

Todos os dados inseridos na interface serão acoplados em duas novas classes, descritas a seguir.

3.2 Novas classes

Para atender aos requisitos que uma simulação de PaaS demanda, foi necessário adicionar duas novas classes ao projeto, uma que represente aplicação e outra para perfis de máquinas virtuais. Ambas as classes pertencem ao escopo de representação icônica no modelo e são totalmente compatíveis com as classes análogas do motor de simulação.

3.2.1 Classe de perfil de máquina virtual

Essa classe armazena informações a respeito de perfis de máquinas virtuais modelados. Um perfil de máquina virtual apenas descreve as características de uma máquina virtual juntamente com o preço seu preço de aquisição. Portanto, um perfil não possui um usuário especifico e deve ser associado a alguma aplicação.

Os atributos que compões essa classe são:

- **Identificador:** Sequencia de caracteres única que identifica cada perfil modelado.
- **Memória primária alocada:** Memória primária que será alocada para a máquina virtual descrita por esse perfil.
- **Memória secundária alocada:** Memória secundária que será alocada para a máquina virtual descrita por esse perfil.
- **Servidor VMM**: Servidor que abriga o *hypervisor* responsável pela alocação escalonamento da máquina virtual criada a partir deste perfil.

- **Sistema Operacional**: Sistema operacional instala na máquina virtual descrita por esse perfil.
- **Custo**: Custo por segundo de cada instancia alocada.
- **Possui *Autoscaling*:** Variável *booleana* que representa se este perfil de máquina virtual implementa *autoscaling*.
- **Tamanho máximo de fila:** Caso esse perfil implemente *autoscaling*, esse atributo define um valor máximo para o tamanho de fila dessa máquina virtual. Caso o tamanho de fila de uma máquina virtual relacionada a esse perfil exceda o valor máximo é solicitado ao Servidor VMM a alocação de mais uma máquina virtual.

3.2.2 Classe de aplicações

Essa classe representa um modelo de aplicações. Como já mencionado, o iSPD enxerga aplicações como um conjunto finito de tarefas, com cargas computacionais e destinos específicos

Essa classe possui os seguintes atributos:

- **Identificador:** Sequencia de caracteres única que identifica cada aplicação.
- **Usuário:** Usuário proprietário da aplicação.
- **Número de tarefas:** Número de tarefas que compõem essa aplicação.
- **Carga computacional máxima:** Valor máximo de carga computacional que uma tarefa pode assumir, em MFlop.
- **Carga computacional mínima:** Valor mínimo de carga computacional que uma tarefa pode assumir, em MFlop.
- **Comunicação máxima:** Valor máximo de comunicação que uma tarefa pode assumir, em *bytes*.
- **Comunicação mínima:** Valor mínimo de comunicação que uma tarefa pode assumir, em *bytes*.
- **Máquina Virtual:** Máquina virtual na qual será executada essa aplicação. Essa máquina virtual possui as características descritas por um modelo de perfil.

3.3 Especificação de escalonadores

Para a caracterização de um serviço de PaaS em no processo de simulação é necessário criar um escalonador específico.

No processo de simulação da classe de serviço IaaS, o operador do simulador define um cluster virtual juntamente com um conjunto de tarefas por usuário. Portanto, essas tarefas são escalonadas por usuários.

Em um simulação de PaaS, é necessário criar uma especialização para esse escalonador, para que ele escalone as tarefas por aplicação.

Essa nova funcionalidade implica que recursos virtuais são provisionados exclusivamente por aplicações e cada cluster virtual apenas escalona tarefas de uma aplicação específica, independente de seu proprietário.

3.4 Provisionamento dinâmico

Em um serviço de PaaS, o provedor tem a responsabilidade de garantir qualidade de serviço condizente com o que foi proposto ao cliente mesmo sem ter conhecimento a cerca do trafego gerado pela aplicação.

Para tanto, é necessário implementar técnicas de provisionamento dinâmico para uma melhor caracterização do serviço de PaaS.

Esta nova funcionalidade consiste em alocar ou desalocar novas máquinas virtuais com base em critérios implementados a partir de políticas específicas para a finalidade

Para realizar essa tarefa, foi elaborada uma nova classe, que representa a interface de cluster virtual de cada aplicação. Essa classe armazena lista de máquinas virtuais disponíveis para uma aplicação específica e aplica as políticas de provisionamento de dinâmico e balanceamento de cargas.

3.4.1 Política baseada em filas

Para realizar o provisionamento dinâmico foi implementada uma nova política específica baseada no tamanho de fila de máquinas virtuais.

Para realizar o *autoscaling*, o operador do simulador define um valor limiar para o tamanho de fila no processo de modelagem de perfis de máquinas virtuais. Quando a fila de uma máquina virtual atinge esse valor uma nova instância é solicitada ao Servidor VMM.

Tendo mais de um nó de máquina virtual por aplicação, a interface de cluster virtual promove o balanceamento de cargas, enviado as novas tarefas ao servidor de menor fila.

Para remover uma instância, a interface de cluster virtual adota o seguinte critério:

Sendo *n* o número, $Fila_i$ o tamanho da fila de um dado nó *i* e *TamMaxFila* o tamanho máximo de fila definido por usuário, tem-se a a condição (4).

$$\frac{\sum_{i=0}^{n} Filai}{n-1} < TamMaxFila \quad (4)$$

Caso a condição (4) for verdadeira, um nó é desalocado.

3.5 Novas métricas

A implementação de novas métricas consiste na construção de uma nova classe que armazene informações a respeito da execução de aplicações.

Essa nova classe possui os seguintes atributos:

- **Identificador:** Uma sequencia de caracteres que idêntica a qual aplicação essa métrica esta relacionada.
- **Usuário:** Usuário proprietário dessa aplicação.
- **Numero máximo de instancias:** Número máximo de máquinas virtuais alocadas para essa aplicação.
- **Custo total:** Custo total de execução da aplicação.
- **Tempo total:** Tempo total de execução da aplicação.

Cada aplicação possui um objeto instanciado desse tipo de classe. Os resultados são exibidos ao final da simulação.

4 TESTES DE NOVAS FUNCIONALIDADES

Para a validação das novas funcionalidades foram realizados testes comparativos com o simulador CloudSim.

O objetivo dos testes é avaliar o provisionamento dinâmico juntamente com a nova política implementada.

Serão consideradas métricas como número máximo de instâncias alocadas, tempo de execução e custos e determinar o impacto da nova política na qualidade de serviço.

4.1 Considerações iniciais

O simulador CloudSim possui um funcionamento semelhante ao iSPD. Cada tarefa, chamada de *cloudlet* possui uma carga computacional e os servidores uma taxa de processamento.

No iSPD a unidade de carga computacional é Mflop e a taxa de processamento Mflops. A documentação do CloudSim define a carga computacional em Mip e a taxa de processamente em Mips.

No entanto, essa situação não vem a ser um empecilho, pois a razão entre a carga de processamento e a taxa de processamento é a mesma ao igualar os valores ignorando as unidades.

4.2 Modelo de ambiente para testes

Para a realização dos testes foi modelado um centro de serviço físico composto por nove máquinas idênticas utilizadas para a hospedagem de máquinas virtuais e um Servidor VMM central.

As máquinas físicas possuem as seguintes características:

- **Taxa de processamento:** 50000 Mflops
- **Núcleos de processamento:** 2
- **Memória primária:** 1024 MB
- **Memória secundária:** 10 GB

O perfil de máquina virtual escolhido para testes possui as seguintes especificações:

- **Núcleos virtuais de processamento:** 1
- **Memória primária:** 1024 MB
- **Memória secundária:** 10 GB
- **Custo:** 1 $/Instância/segundo

A taxa de processamento da máquina virtual será a mesma do servidor físico que a hospeda. Como todos servidores são idênticos, a taxa será de 50000 Mflops.

O perfil definido também usará *autoscaling* e o tamanho máximo de fila será definido de acordo com o contexto.

Na Figura 4.6 o modelo físico é ilustrado a partir da interface gráfica do iSPD.

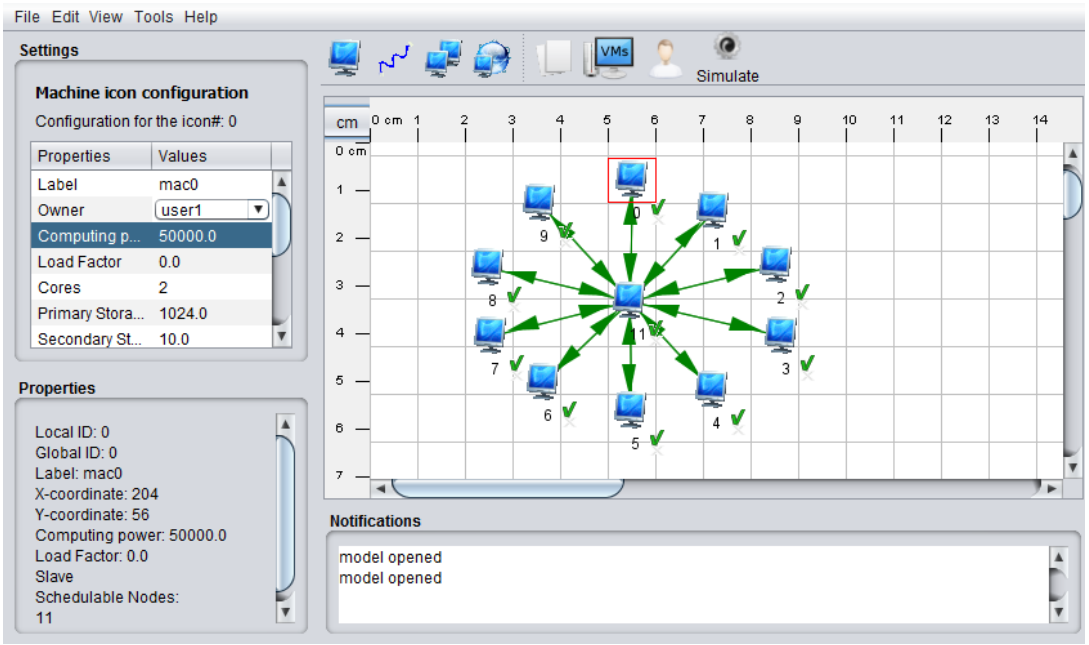

Figura 4.6: Modelo de testes representado no iSPD

4.3 Realização e Resultados

Foram realizados testes abordando dois tipos de situações:

- **Alto tráfego:** As aplicações possuem um grande número de tarefas, no entanto com baixa carga computacional.
- **Baixo tráfego:** As aplicações possuem um baixo número de tarefas, no entanto apresentam alta carga computacional.

Cada uma dessas situações foi divida em duas seções com dez iterações por teste. A primeira seção avalia as métricas usando provisionamento dinâmico variando o número de tarefas em cada iteração. A segunda realiza um teste análogo usando provisionamento estático, ou seja, um número fixo de máquinas virtuais.

4.3.1 Primeira seção de testes de alto tráfego

Nesse teste cada tarefa tem uma carga computacional de 100000 Mflop e o limiar estabelecido para solicitação de *autoscaling* foi de 2000 tarefas enfileiradas em apenas uma máquina virtual.

Na primeira iteração do teste a aplicação possuía 3500 tarefas e a cada iteração esse número foi incrementado em 1000 unidades.

Na Tabela 4.1 estão os resultados obtidos na simulação realizada pelo iSPD e na Tabela 4.2 os resultados da simulação realizada no CloudSim.

Ambas as tabelas exibem as seguintes métricas:

- **Número máximo de instâncias utilizadas**
- **Tempo total de processamento em segundos**
- **Custo de implantação**

Tabela 4.1: Resultados da primeira seção de testes de alto tráfego no iSPD.

Iteração	Tarefas	Instâncias	Tempo	Custo
1	3500	2	4008	8011,81
2	4500	3	4014	12025,12
3	5500	3	4010	11001,76
4	6500	4	4016	16040,86
5	7500	4	4012	16029,92
6	8500	5	4018	20057,55
7	9500	5	4012	19008,28
8	10500	6	4018	21001,02
9	11500	6	4014	23001,96
10	12500	7	4020	25002,11

Tabela 4.2: Resultados da primeira seção de testes de alto tráfego no CloudSim.

Iteração	Tarefas	Instâncias	Tempo	Custo
1	3500	2	4006	8012,10
2	4500	3	4006	12018,10
3	5500	3	4006	12018,10
4	6500	4	4006	16024,10
5	7500	4	4006	16024,10
6	8500	5	4006	20030,10
7	9500	5	4006	20030,10
8	10500	6	4006	24036,10
9	11500	6	4006	24036,10
10	12500	7	4006	28042,10

Percebe-se que ao aumentar o número de tarefas aumenta-se o número de instâncias e consequentemente o custo total de implantação da aplicação. No entanto, o tempo total conserva-se, evidenciando que houve uma conservação na qualidade de serviço proporcionada pela política de provisionamento dinâmico implementada.

Na Tabela 4.3 são exibidos os valores médios de tempo obtidos com seus respectivos desvios padrão para cada simulador.

Tabela 4.3: Tempo médio e desvio padrão relativos a primera seção de testes de alto tráfego.

Simulador	Tempo médio	Desvio padrão
iSPD	4012	4
CloudSim	4006	0

As variações do número de instâncias em função do número de tarefas também coincidiram em ambos os compiladores. Na Figura 4.7 é representado o gráfico do número de instâncias em função do número de tarefas propiciado pela política de provisionamento dinâmico implementada.

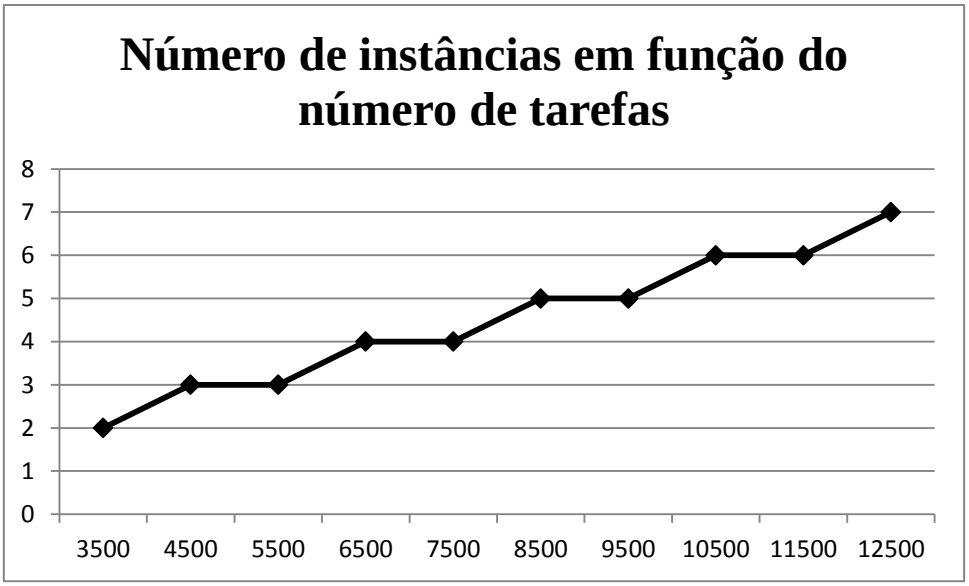

Figura 4.7: Gráfico de variação do número de instâncias em função do número de tarefas em alto tráfego.

Usando método de regressão linear, obtemos o seguinte valor para a função de variação do número de instâncias, em que *n* representa o número de instâncias e *x* o número de tarefas:

$$n = \lceil 0{,}00052 * x \rceil \quad (4)$$

4.3.2 Segunda seção de testes de alto tráfego

Nesta seção, o número de instâncias em cada iteração corresponde ao mesmo número de instâncias provisionadas dinamicamente na primeira seção para cada teste realizado.

No entanto, não será levado em conta o custo, apenas o tempo total de execução da aplicação, uma vez que a política de cobrança difere de um serviço para outro.

A seguir, na Tabela 4.4 estão representados o resultados obtidos em relação ao simulador iSPD e na Tabela 4.5 os resultados obtidos em relação ao simulador CloudSim.

Tabela 4.4: Segunda seção de testes para alto tráfego no iSPD.

Iteração	Tarefas	Instâncias	Tempo
1	3500	2	3500
2	4500	3	3000
3	5500	3	3668
4	6500	4	3250
5	7500	4	3750
6	8500	5	3400
7	9500	5	3800
8	10500	6	3500
9	11500	6	3834
10	12500	7	3572

Tabela 4.5: Segunda seção de testes para alto tráfego no CloudSim.

Iteração	Tarefas	Instâncias	Tempo
1	3500	2	3500
2	4500	3	3000
3	5500	3	3668
4	6500	4	3250
5	7500	4	3750
6	8500	5	3400
7	9500	5	3800
8	10500	6	3500
9	11500	6	3834
10	12500	7	3572

Na Tabela 4.6 são exibidas as médias e desvio padrões relativos aos tempos médios obtidos nos testes.

Tabela 4.6: Comparativo entre tempo de diferentes politicas de provisionamento em alto tráfego

Provisionamento	Tempo (iSPD)	Tempo (CloudSim)	Desvio Padrao (iSPD)	Desvio padrão (CloudSim)
Dinâmico	4012	4006	4	0
Estático	3527	3527	260	260

Neste caso, o provisionamento estático não apresentou um tempo constante, no entanto, para todas as iterações, apresentou um melhor desempenho do que o provisionamento dinâmico.

4.3.3 Primeira seção de testes de baixo tráfego

Neste teste, as tarefas possuem 100000000 Mflop de carga de computação e um limiar de 10 unidades de processos enfileirados em uma única máquina virtual para que seja solicitado *autoscaling*.

Na primeira iteração a aplicação possuía 15 tarefas e a cada iteração esse número foi incrementado em 10 unidades.

Na Tabela 4.7 são exibidos os resultados obtidos a partir do simulador iSPD e na Tabela 4.8 os resultados obtidos a partir do simulador CloudSim.

Tabela 4.7: Resultados da primeira seção de testes de baixo tráfego no iSPD.

Iteração	Tarefas	Instancias	Tempo	Custo
1	15	2	24000	47990,36
2	25	3	24004	71993,24
3	35	3	24002	70000,34
4	45	4	24003	90003,66
5	55	5	24004	110002,86
6	65	6	24008	130007,17
7	75	7	24008	150003,85
8	85	8	24010	191993,58
9	95	8	24007	190001,66
10	105	9	24009	210001,42

Tabela 4.8: Resultados da primeira seção de testes de baixo tráfego no CloudSim.

Iteração	Tarefas	Instancias	Tempo	Custo
1	15	2	24000	48000,10
2	25	3	24000	72000,10
3	35	3	24000	72000,10
4	45	4	24000	96000,10
5	55	5	24000	120000,10
6	65	6	24000	144000,10
7	75	7	24000	168000,10
8	85	7	24000	168000,10
9	95	8	24000	192000,10
10	105	9	24000	216000,10

Como no teste de alto tráfego, a política de provisionamento dinâmico proporcionou uma conservação do tempo de execução total da aplicação aumentando o número de instancias e consequentemente custo de implantação.

Na Tabela 4.9 são exibidos os valores médios de tempo obtidos com seus respectivos desvios padrão para cada simulador.

Tabela 4.9: Tempo médio e desvio padrão relativos a primeira seção de testes de baixo tráfego.

Simulador	Tempo médio	Desvio padrão
iSPD	24006	3
CloudSim	24000	0

O número de instâncias coincidiu para ambos os compiladores novamente. O gráfico representado na Figura 4.8 descreve a variação de instâncias por quantidade de tarefas.

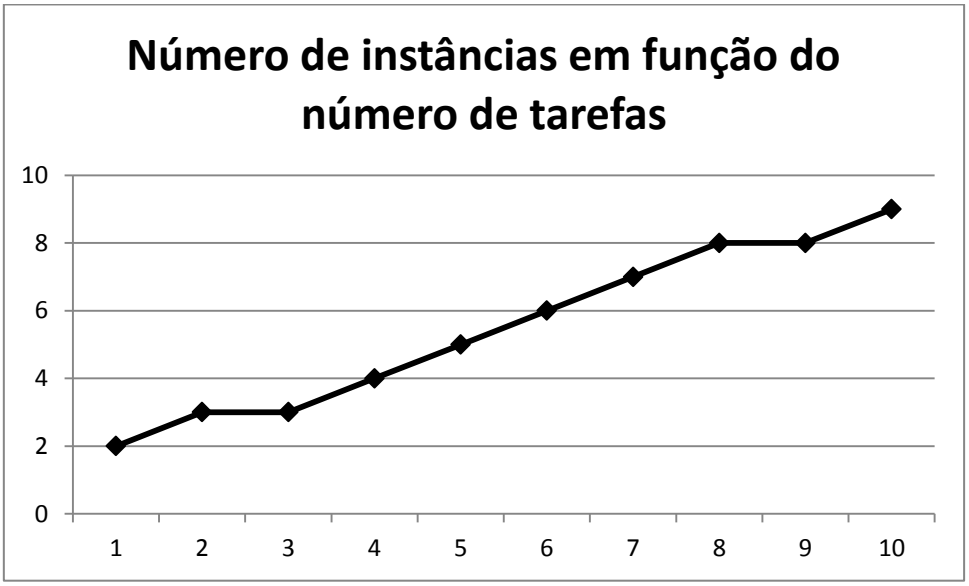

Figura 4.8: Gráfico de variação do número de instâncias em função do número de tarefas em baixo tráfego.

Usando regressão linear, obtém-se a função (5), em que *n* representa o número de instâncias e *x* o número de tarefas:

$$n = \lceil 0{,}0806 * x \rceil \quad (5)$$

4.3.4 Segunda seção de testes de baixo tráfego

Assim como foi realizado para aplicações de alto tráfego, esta seção de testes apresenta um comparativo entre a política de provisionamento dinâmico com o provisionamento estático. Os testes a seguir usam um número fixo de máquinas virtuais, que correspondem ao número apresentado em cada iteração do teste passado.

Na Tabela 4.10 são exibidos os resultados obtidos a partir do simulador iSPD e na Tabela 4.11 os resultados obtidos a partir do simulador CloudSim.

Tabela 4.10: Resultados da segunda seção de testes de baixo tráfego no iSPD.

Iteração	Tarefas	Instancias	Tempo
1	15	2	16000
2	25	3	18000
3	35	3	24000
4	45	4	24000
5	55	5	22000
6	65	6	22000
7	75	7	22000
8	85	8	22000
9	95	8	24000
10	105	9	24000

Tabela 4.11: Resultados da segunda seção de testes de baixo tráfego no CloudSim.

Iteração	Tarefas	Instancias	Tempo
1	15	2	16000
2	25	3	18000
3	35	3	24000
4	45	4	24000
5	55	5	22000
6	65	6	22000
7	75	7	22000
8	85	8	22000
9	95	8	24000
10	105	9	24000

Na Tabela 4.12 é realizado o comparativo entre tempos médios e desvios padrões das diferentes políticas de provisionamento.

Tabela 4.12: Comparativo entre tempo de diferentes politicas de provisionamento em baixo tráfego

Provisionamento	Tempo (iSPD)	Tempo (CloudSim)	Desvio Padrao (iSPD)	Desvio padrão (CloudSim)
Dinâmico	24006	24000	3	0
Estático	21800	21800	2740	2740

Novamente, ocorre uma grande discrepância entre os valores apresentados em cada iteração para a política de provisionamento estático, mostrando que não há conservação de tempo. No entanto, ainda para o baixo tráfego, a política de provisionamento estático apresentou menor tempo de execução.

4.4 Considerações finais

Em todos os testes realizados o simulador iSPD apresentou grande similaridade de resultados com o simulador CloudSim. Quanto à análise da política de provisionamento dinâmico ficaram evidentes alguns aspectos:

- **Conservação de qualidade de serviço:** Ao aumentar o número de tarefas, aumenta-se a demanda por poder computacional. A política implementada foi capaz de manter o tempo total de execução da aplicação independente do número de tarefas submetidas, devido a sua capacidade de provisionar novos recursos, conferindo elasticidade rápida ao serviço oferecido.

- **Proporção de custo:** Ao aumentar a demanda por poder computacional, aumenta-se a quantidade de recursos disponíveis para a aplicação. O custo total de implantação é proporcional aos recursos provisionados, a fim de se manter a qualidade de serviço oferecida pelo provedor.

No entanto, ao comparar a política de provisionamento dinâmico com o provisionamento estático observa-se no tempo total de execução. A política de provisionamento estático não apresentou conservação do tempo total de execução ao incrementar o número de tarefas. No entanto, em todas as iterações, o tempo total de execução dessa política foi menor que o tempo total de execução da política de provisionamento dinâmico.

Para tanto, o número de máquinas virtuais inseridas nos testes de provisionamento estático foram determinados nos testes anteriores com a política de provisionamento dinâmico.

Em suma, o provisionamento estático se mostrou mais eficiente, no entanto, é necessário que se conheça a demanda computacional da aplicação implantada, diferente do caso de políticas de provisionamento dinâmico, onde o sistema deve lidar com aumentos de tráfego não previstos.

5 CONCLUSÕES

O processo de simulação pode contribuir significativamente com o desempenho de um sistema. A discussão proposta nesse trabalho mostra que escolhendo a estratégia adequada é possível alcançar melhores resultados, implicando em economia de tempo e dinheiro.

A política de provisionamento dinâmica implementada no iSPD realizar este trabalho não é nativa no simulador CloudSim, portanto, foi necessária a implementação da mesma no simulador para a realização de testes comparativos.

Em suma, o iSPD é uma novidade entre os simuladores de computação em nuvem. Além de oferecer uma interface gráfica de fácil utilização, é implementada a camada serviço, tal como IaaS e PaaS, diferente dos demais simuladores, os quais apenas simulam a infraestrutura do sistema de computação em nuvem.

5.1 Ações futuras

O iSPD ainda não contempla a simulação da classe de serviços SaaS, o que pode vir a se tornar um trabalho futuro.

Provisionamento dinâmico esta presenta em todas as classes de serviço de computação em nuvem, portanto, é possível implementar novas políticas concomitante a implementação da classe de serviço SaaS.

Referências

BUYYA, R.; BROBERG, J.; GOSCINSKI, A. Cloud Computing: Principles and Paradigms. Wiley, 2011. (Wiley Series on Parallel and Distributed Computing). ISBN 9781118002209.

RITTINGHOUSE, J.; RANSOME, J. F. Cloud Computing: Implementation, Management, and Security. [S.l.]: CRC, 2009.

BUYYA, R.; RANJAN, R.; CALHEIROS, R. N.; Modeling and simulation of scalable cloud computing environments and the Cloudsim toolkit: Challenges and opportunities. CoRR, abs/0907.4878, 2009.

CASTANE, G. G.; NUNEZ, A.; CARRETERO, J. iCanCloud: A brief archtecture overview. In: ISPA. IEEE, 2012. p. 853-854. ISBN 978-1-4673-1631-6.

KLIAZOVICH, D. et al. GreenCloud: A packet-level simulator of energy-aware cloud computing data centers. In: Global Telecommunications Conference (GLOBECOM 2010), 2010. IEEE [S.I.: s.n.], 2010 p. 1-5. ISSN 1930-529X.

MANACERO, A.; LOBATO, R.S.; OLIVEIRA, P.H.M.A; GARCIA, M.A.B.A; GUERRA A.I.; AOQUI,V. ; MENEZES D.; SILVA, D.T; iSPD: an iconic-based modeling simulator for distributed grids. In: Proc. of 45th Annual Simulation Symposium (ANSS'12), Orlando, EUA, 2012, p.1-8

SILVA, D. T., MANACERO, A., JORGE, A., MENEZES, D., e LOBATO, R. S. e SPOLON, R. Simulação de sistemas de computação em nuvem para o iSPD. Revista Interciência & Sociedade, 2015, v. 4, p. 86–93.

MELL, P.; GRANCE, T. The NIST Definition of Cloud Computing. Gaithersburg, MD, 2011.

XEN Project. Disponível em <http://www.xenproject.org/>. Acessado em 2015.

VMWare. Disponível em <http://www.vmware.com/br>. Acessado em 2015.

LORIDO-BOTRÁN, T.; MIGUEL-ALONSO, J.; LOZANO, J.A.; Auto-scaling techniques for elastic applications in cloud environments. Department of Computer

Architecture and Technology, University of Basque Country, Tech. Rep. EHU-KAT-IK-09, v. 12, p. 2012, 2012.

HUNG, C.; HU, Y.; LI, K.; Auto-Scaling Model for Cloud Computing System. International Journal of Hybrid Information Technology, v. 5, n. 2, p. 181-186, 2012.

App Engine. Disponível em <https://cloud.google.com/appengine>. Acessado em 2015.

Microsoft Azure. Disponível em <https://azure.microsoft.com/pt-br>. Acessado em 2015.

OMNET++. Discrete event simulator. Disponível em < https://omnetpp.org/>. Acessado em 2015

BELOGLAZOV, A.; BUYYA, R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. Concurr. Comput. : Pract. Exper., John Wiley and Sons Ltd., Chichester, UK, v. 24, n. 13, p. 1397–1420, set. 2012. ISSN 1532-0626.