UNIVERSIDADE FEDERAL FLUMINENSE

CARLOS HENRIQUE ZILVES NICODEMUS
DANIEL DE SOUZA PINTO

SGCCloud: Uma Proposta para Gerenciamento e Implementação de Computação nas Nuvens

NITERÓI

2010

UNIVERSIDADE FEDERAL FLUMINENSE

CARLOS HENRIQUE ZILVES NICODEMUS
DANIEL DE SOUZA PINTO

SGCCloud: Uma Proposta para Gerenciamento e Implementação de Computação nas Nuvens

Trabalho submetido ao Curso de Bacharelado em Ciência da Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação. Área de concentração: Computação Distribuída.

Orientador:
Prof. PhD. EUGENE FRANCIS VINOD REBELLO

NITERÓI

2010

SGCCloud: Uma Proposta para Gerenciamento e Implementação de Computação nas Nuvens

Carlos Henrique Zilves Nicodemus
Daniel de Souza Pinto

Trabalho submetido ao Curso de Bacharelado em Ciência da Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Aprovada por:

Prof. PhD. Eugene Francis Vinod Rebello / IC-UFF
(Presidente)

Profa. PhD. Maria Cristina Silva Boeres / IC-UFF

D.Sc. Alexandre Sena / LASALLE-RJ

Niterói, 12 de julho de 2010.

"Escolhe um trabalho de que gostes,
e não terás que trabalhar nem um dia na tua vida."
Confúcio

Agradecimentos

Agradeço a Deus, pela graça alcançada.
As nossas famílias, que nos apóiam em todas as etapas de nossas vidas.
Ao professor e orientador Vinod por seu apoio e inspiração no amadurecimento dos nossos conhecimentos e conceitos que nos levaram a execução e conclusão desta monografia.
A todos os amigos do SGCLab pelo incentivo e apoio dados durante estes muitos anos de convivência.

Resumo

As crescentes buscas por poder computacional nas diversas áreas científicas e empresariais aqueceram as pesquisas na área de computação paralela e distribuída. Várias propostas de modelo de computação paralela foram apresentadas por estudiosos e empresas do ramo da computação.

A computação em nuvens se apresenta como uma solução para prover uma grande quantidade de poder computacional. Apesar de ser um conceito ainda abstrato, ela é considerada como uma quebra de paradigma e o próximo passo na evolução da computação como serviço. Realmente por suas características a computação em nuvens se mostrou um modelo bem promissor com características bem atraentes e promessas de computação barata e infinita.

Esse trabalho propõe uma discussão sobre os requisitos necessários para a computação em nuvens e entender um pouco melhor sobre seus conceitos. Foi feito também um protótipo composto de um *middleware* e um portal de acesso a fim de entender as dificuldades no gerenciamento das máquinas físicas e no oferecimento de recursos virtuais para os usuários finais de forma simples e eficiente.

Palavras-chave: computação distribuida, computação em nuvens, virtualização, VMWare, middleware, portal, Java, GWT

Abstract

The increasing searches for computational power in different areas of science and commerce has heated up research on parallel and distributed computing. Various proposals for parallel computing models have been presented by scholars and companies in the business of computing.

The cloud computing is being presented as the latest solution that promises to provide a large amount of computing power. Although still an abstract concept, cloud computing is considered a paradigm shift and the next step on the evolution towards computing as a service. At this early stage, and with some attractive features, the cloud computing model is already beginning deliver on its promise of cheap and infinite computing.

This work a discuss the basic requirements for a cloud in order for the reader to understand this concept a little better. A prototype was also made, consisting of a middleware and a portal in order to presents the difficulties of managing the physical machines that provide virtual resources for end users in a simple and efficient way.

Keywords: distributed computing, cloud computing, virtualization, VMware, middleware, portal, Java, GWT

Palavras-chave

1. Computação nas Nuvens
2. Virtualização
3. Grades Computacionais
4. Computação Utilitária
5. Desenvolvimento de Portais

Glossário

VMM	:	Virtual Machine Monitor;
SSL	:	Security Socket Layer;
NFS	:	Network File System;
NIS	:	Network Information Service;
GWT	:	Google Web Toolkit;
RPC	:	Remote Procedure Call ;
RMI	:	Remote Method Invocation;
TI	:	Tecnologia da Informação;
UFF	:	Universidade Federal Fluminense;
IC	:	Instituto de Computação;

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O grande motivo que impulsiona idéias como a da computação em nuvens (*cloud computing*) é a mesma que movimenta toda a indústria de computação: oferecer cada vez mais poder computacional. Com o passar dos anos viu-se que a taxa de crescimento do poder computacional não seria suficiente para suprir as diversas áreas científicas e empresariais (RODRIGUES, 2009). Arquiteturas computacionais mais robustas que fossem capazes de atender essa necessidade seriam impossíveis de serem projetadas ou demandariam um investimento em hardware e manutenção inviáveis. Dessa demanda surgiram vários projetos que propunham a junção de poder computacional de máquinas simples para formar uma máquina virtualmente capaz de atender essa demanda (BOERES et al., 2006).

Podemos citar alguns projetos que se baseiam nessa idéia da junção de poder computacional: o SETI@HOME (ANDERSON et al., 2002) agrega o tempo livre de computação dos internautas para pesquisar vida extraterrestre inteligente; o *World Community Grid* (WORLD, 2010) que tenta criar o maior supercomputador público do mundo para ser usado em pesquisas científicas; o *Climate Prediction* (STAINFORTH et al., 2002) que é uma grade para tentar predizer as mudanças climáticas ao redor do mundo até 2100. Esses são somente alguns projetos científicos onde sem a computação distribuída não haveria sucesso.

A computação em nuvens é um paradigma que vem sendo amplamente discutido tanto por instituições acadêmicas quanto por empresas privadas. Porém, apesar de várias empresas já oferecerem serviços de computação em nuvens em várias vertentes, ainda não se chegou a um consenso sobre uma definição formal e definitiva sobre essa nova solução para oferecer poder computacional como um serviço.

Em 2009 foi publicado o *Open Cloud Manifesto* (OPEN, 2009), documento que tinha

como objetivo formalizar a discussão sobre o assunto. Apesar de não detalhar definições ou formas de implementação, a discussão sobre nuvens computacionais públicas chamou a atenção de grandes empresas do ramo como a Microsoft e a Amazon. Nesse contexto pode-se dimensionar a importância desse paradigma no meio comercial.

1.1 Motivação e Objetivo

A computação em nuvens promete oferecer uma forma de computação totalmente flexível e capaz de prover todo o poder computacional que for necessário para o usuário como descrito em (ARMBRUST et al., 2009). Para tal, torna-se necessário uma estrutura de gerenciamento dos recursos, dos dados e dos processos do usuário que seja transparente, simples e eficiente.

O laboratório SGCLab vem pesquisando técnicas de programação paralela e distribuída, publicando vários trabalhos e mantendo linhas de pesquisa principalmente na área de grades computacionais. O objetivo deste trabalho foi tentar aprofundar um pouco mais nos requisitos e no funcionamento dessa complexa estrutura e estudar alternativas para solucionar alguns dos desafios impostos. Utilizando o conhecimento e o trabalho já desenvolvido na infra-estrutura da grade e propondo algumas soluções para formar uma estrutura de nuvem computacional.

Para tal construímos o SGCCloud, um protótipo formado por um portal e um *middleware* para gerenciamento e acesso a uma infra-estrutura de computação em nuvens, que são peças chaves para atender aos requisitos do modelo. Através desse protótipo pode-se fazer algumas observações e entender um pouco melhor as dificuldades de implementação. Espera-se também que esse trabalho seja a base para projetos futuros que resultem na real implementação de uma infra-estrutura de computação em nuvens completa de acordo com os ideais de computação distribuída do laboratório SGCLab.

1.2 Organização

Este trabalho está estruturado da seguinte forma. O capítulo dois aborda os conceitos de grade computacional (*grid computing*) e computação utilitária (*utility computing*) que são conceitos bem semelhantes ao de computação em nuvens e que contribuíram bastante para sua formulação.

O capítulo três define o conceito de computação em nuvens que será usado como base durante o trabalho, e é nesse conceito que será baseado o projeto da arquitetura do SGCCloud.

O capítulo quatro apresenta os conceitos de virtualização, tipos, propriedades e vantagens. É um dos fundamentos utilizados na implementação de uma infra-estrutura de computação em nuvens no SGCCloud.

A organização e a implementação do protótipo proposto são apresentadas no capítulo cinco. Também são abordadas as dificuldades enfrentadas no desenvolvimento do SGCCloud.

Por fim, no capitulo seis são apresentadas as conclusões e propostas para soluções alternativas aos problemas citados que deverão ser solucionados em trabalhos posteriores.

Capítulo 2

Grades e Computação Utilitária

Grades computacionais (*grid computing*) (FOSTER, 2001) e Computação Utilitária (*utility computing*) (YEO et al., 2006) são dois conceitos que possuem características herdadas pelo paradigma da computação em nuvens. Por esse motivo muitas vezes esses paradigmas são confundidos e se torna difícil classificar o que seria uma nuvem computacional.

Para sanar algumas dúvidas e mostrar a relação existente entre esses modelos e a nuvem computacional, são abordadas essas duas metodologias mostrando as suas respectivas contribuições na construção do conceito de computação em nuvens.

2.1 Grades Computacionais

Denomina-se como grade computacional a junção de poder computacional entre várias máquinas, geralmente geograficamente distribuídas, com a finalidade de resolver um problema específico. As máquinas da grade possuem as características de serem heterogêneas e interligadas ao redor do planeta através da internet.

Segundo (NASCIMENTO, 2006), a característica que promoveu o sucesso do sistema de grades computacionais é a capacidade de oferecer o desempenho necessário para resolver um problema para qualquer pessoa em qualquer parte do mundo através da internet. Contudo, para viabilizar a utilização da grade é necessário o desenvolvimento de uma série de ferramentas para gerência e tomada de decisões que facilitem a utilização de um usuário sem o conhecimento específico para operar a grade.

A figura 2.1 exemplifica como pode ser a distribuição geográfica dos recursos que

compõem uma grade computacional. Cada computador no mapa representa um conjunto local de recursos computacionais e as ligações entre eles são feitas através da internet.

Figura 2.1: Exemplo de uma grade computacional.

Para o sucesso da grade é importante que seus usuários não vejam a infra-estrutura em si, e sim uma máquina virtual robusta capaz de rodar seu algoritmo distribuído em um tempo que o mesmo considera razoável, de uma forma barata e garantindo um grau de segurança esperado.

2.1.1 Obstáculos da Abordagem

O ambiente da grade apresenta diversas variáveis que precisam ser gerenciadas e contabilizadas em algumas tomadas de decisão. O fato de ser um ambiente formado por máquinas heterogêneas (FOSTER, 2001), por exemplo, deve ser levado em conta na escolha de qual a melhor máquina para alocar uma tarefa. Tarefas que exigem muito processamento devem ser alocadas em máquinas com processadores mais ociosos. Já tarefas que usem muito memória devem ser associadas com máquinas com mais memória disponível.

A infra-estrutura de comunicação também é um grande problema. Como as máquinas se comunicam através da internet, há uma variância entre a latência de comunicação entre as mesmas, e a ocorrência de falhas é bem frequente. Um escalonador deve escolher as máquinas levando em consideração também o tempo de comunicação entre os processos, além de haver uma tolerância a falhas para se tomar uma decisão no caso de perda de conexão com um dos nós da grade. Esse mecanismo de tolerância a falhas também deve

garantir a integridade do processamento no caso de uma das máquinas cair ou se tornar indisponível por qualquer motivo.

Para garantir toda a abstração na utilização da grade, um *middleware* e um portal também são necessários. O *middleware* atua na comunicação com a infra-estrutura de *hardware* e abstrai as tarefas como alocar o recurso, fazer o escalonamento, transferir os arquivos necessários, etc. O portal é o ponto de acesso do usuário na grade. É através dele que o usuário irá efetivamente interagir e aproveitar os recursos da grade.

2.1.2 Principais diferenças e contribuições da grade para a computação em nuvens

A grande característica da computação em nuvens é a escalabilidade sob demanda. Um sistema de nuvem computacional deve alocar recursos e desalocá-los conforme a necessidade do cliente. Grades computacionais possuem essa mesma característica, elas devem oferecer o poder computacional necessário para atender as necessidades do usuário em um determinado instante. Contudo, depois que o recurso da grade é alocado ele só será desalocado depois do término da execução da tarefa, e novos recursos não podem ser adicionados durante a execução. Nesse ponto a nuvem computacional é mais flexível. O usuário faz a requisição da quantidade de recurso que ele quer inicialmente e pode optar por ter mais ou menos recursos a qualquer momento.

O problema e solução do balanceamento de cargas foi também inserido inicialmente pelo paradigma de grades. Um gerenciador capaz de tomar decisões de balanceamento de cargas visando maximizar a utilização dos seus recursos. Esse é um problema também da computação em nuvens, uma grande infra-estrutura deve ser capaz de distribuir bem os processos pelos seus recursos, e várias idéias antes implementadas para grades foram reutilizadas na computação em nuvens. A principal é que nuvens computacionais são utilizadas para executar múltiplas aplicações simultaneamente sob a mesma infra-estrutura, isto é, aplicações teriam que compartilhar recursos físicos.

Há também inúmeras outras contribuições que o paradigma de computação em nuvens herdou das grades computacionais. Como, por exemplo, a necessidade do próprio *middleware* e do portal como camada de abstração para os usuários, os algoritmos de tolerância a falhas responsáveis por manter a confiabilidade do processamento, os gerenciadores de aplicação e *hardware*, dentre outras.

Contudo a principal diferença entre as duas metodologias está em suas finalidades.

As duas foram idealizadas para oferecer recursos como serviço, entretanto grades foram montadas como uma alternativa colaborativa onde instituições oferecem seus recursos para a grade enquanto usam recursos da grade. Grades computacionais geralmente são montadas com uma intenção específica, agregando recursos, geralmente de instituições educacionais, para resolver problemas de um grupo específico e logo depois de ter cumprido sua premissa são desfeitas. Por outro lado, nuvens computacionais tem um apelo mais comercial. A idéia é prover poder computacional e de armazenamento para quem precisar, cobrando pelo que foi usado. Geralmente não é geograficamente distribuído e a base de sua infra-estrutura é um *datacenter* pertencente a uma empresa, que aluga seus recursos na tentativa de minimizar o desperdício de poder computacional quando sua necessidade de poder computacional é baixa.

2.2 Computação Utilitária

A computação se tornou um componente essencial na sociedade moderna. A cada atividade que fazemos há sistemas computacionais presentes para auxiliar ou tornar certas ações possíveis. Foi esse contexto que propiciou a idéia de prover computação como um serviço. Computação Utilitária é um conceito que tem como objetivo oferecer a computação como um serviço básico, assim como energia elétrica, água, telefone, etc.

Ela se baseia no fato de que como todo serviço, depois que um usuário passa por um período de adaptação e descobre novas formas de usar o serviço em benefício do seu dia-a-dia, a computação também se torna transparente e indispensável. Hoje já podemos ver claramente que o caminho da computação é realmente esse. Cada vez mais pessoas possuem computadores pessoais em casa ou possuem computadores portáteis. Celulares estão cada vez mais complexos e acumulando várias funções, tornando-se uma ferramenta que é mais que um simples telefone móvel. Serviços como movimentações bancárias, distribuição de notícias, correio eletrônico, *chat* já fazem parte da rotina de um grupo cada vez maior de pessoas.

A idéia da computação utilitária é exatamente oferecer a computação como um serviço. Computação nesse caso pode ser via *software*, armazenamento, processamento ou qualquer outra funcionalidade computacional que traga benefícios ao usuário. Assim, como todo serviço, o poder computacional seria cobrado de acordo com que o usuário usa, dando mais apelo comercial à computação.

As vantagens desse modelo são o melhor aproveitamento dos recursos, visto que em

datacenters, grande parte do tempo os recursos ficam ociosos porque os requisitos mínimos de uma infra-estrutura de *hardware* são montados visando atender aos horários de pico de utilização; economia, compra de hardware e investimentos de manutenção e configuração, ao invés de bancar e manter um *datacenter* inteiro só para atender à sua maior demanda de poder computacional, paga-se somente o que for usado de uma infra-estrutura alugada; novas facilidades aos usuários através de *softwares* fornecidos pela internet que podem ser acessados de qualquer computador ou dispositivo móvel, *internet banking*, *home brokers* e tantos outros serviços que se encontram a disposição.

Contudo, como todo serviço público prestado, o mesmo deve estar prontamente disponível sempre que necessário. A perda intermitente do serviço pode causar sérios problemas e inconvenientes para o usuário, portanto nenhuma falha nesse sistema é aceitável. Esse é um grande obstáculo a se transpor, pois prover um serviço computacional protegido contra falhas é uma tarefa árdua. Outro ponto igualmente importante é a questão da segurança, para obter a confiança dos usuários é necessário garantir que as informações confiadas estejam totalmente protegidas contra pessoas maliciosas. Então várias camadas e protocolos de segurança são indispensáveis para manter o serviço seguro e confiável.

Embora o conceito de computação utilitária seja muito semelhante com o de computação em nuvens, devemos esclarecer as diferenças entre elas. A primeira dita um modelo de negócios onde a computação é um produto com grande potencial de sucesso para o mercado devido as suas vantagens. Enquanto a computação em nuvens trata de como projetar, construir, implantar e executar aplicações que operam em um ambiente virtualizado para comercializar ou simplesmente contribuir e usufruir de uma estrutura com poder computacional grandioso.

Capítulo 3

Computação em Nuvens

Como já falamos, existem várias definições para computação em nuvens. Nesse trabalho adotou-se que computação em nuvens (OPEN, 2009) é o compartilhamento de recursos computacionais sob demanda através da internet sem transferir para o usuário final a complexidade de gerenciamento de uma infra-estrutura complexa. Entretanto, outra definição interessante consiste em computação em nuvens como sendo um paradigma de computação distribuída de larga escala e que é regida por razões econômicas, onde um conjunto abstrato, virtual, escalável dinamicamente e auto-gerenciado de poder computacional, armazenamento, plataforma e serviço é oferecido sob demanda para clientes externos pela internet (FOSTER et al., 2008).

"Várias pessoas estão entrando na nuvem, mas eu não ouvi duas pessoas dizerem a mesma coisa sobre ela. Existem várias definições fora da nuvem."
Andy Isherwood, ZDnet News, 11 de dezembro de 2008

A estrutura de uma nuvem computacional oferece todo o gerenciamento, configurações, *hardwares* e suporte de forma transparente para atender as necessidades de poder computacional e de serviços exigida pela crescente indústria de TI, sem envolver o usuário nessa complexa estrutura. Isso significa que o usuário não precisa se preocupar com questões como gerenciamento de *hardware*, *software*, rede ou limitações de processamento e armazenamento, podendo dar foco na qualidade de suas outras atribuições.

De acordo com a distribuição de recursos, computação em nuvens pode ser classificado como: (ZHANG; CHENG; BOUTABA, 2010)

- *Infrastructure as a Service* (IaaS): quando se utiliza uma porcentagem de um servidor, geralmente com configuração que se adéque à sua necessidade. Exemplos:

firewall e armazenamento.

- *Plataform as a Service* (PaaS): utilizando-se apenas uma plataforma como um banco de dados, um servidor *web*, um *desktop* virtual, etc. Exemplos: Amazon Elastic Compute Cloud (EC2) e API's do Google como o *Gears* e *App Engine*.

- *Development as a Service* (DaaS): as ferramentas de desenvolvimento tomam forma na computação em nuvens como ferramentas compartilhadas, ferramentas de desenvolvimento baseadas em *web* e serviços baseados em *mashup* (uma aplicação montada através de funcionalidades existentes em um banco de funcionalidades). Exemplo: A IDE *online The Force* para desenvolvimento a partir de um navegador.

- *Software as a Service* (SaaS): uso de um software em regime de utilização *web*. Exemplo: *software* oferecidos na internet feitos geralmente em Ajax e ASP.

Na figura 3.1, mostra o que cada tipo de computação em nuvens oferece, desde serviços básicos de infra-estrutura como firewall e armazenamento até serviços para execução de aplicações remotas.

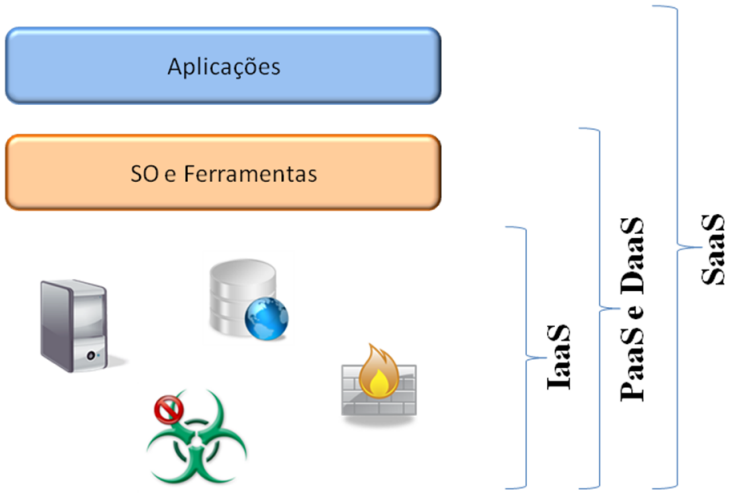

Figura 3.1: Modelo com os tipos de computação em nuvens.

O mercado tem apresentado um respaldo na disseminação desse paradigma, oferencendo facilidades e oportunidades às empresas de ter alto poder computacional por um custo inicial extremamente reduzido ou nulo. Sua popularização se deu também por servir o mercado em basicamente duas frentes diferentes: a computação utilitária e o *Software as a Service (SaaS)*.

A computação utilitária visa oferecer ao público poder de processamento ou armazenamento como um serviço. Esse sistema tem como principal vantagem o baixo custo

inicial com *hardware* necessário para montagem da infra-estrutura requerida. Porém podemos interpretar como uma desvantagem o fato do *hardware* ser totalmente alugado ou emprestado e que algumas variáveis podem ficam fora de controle.

Hoje as empresas que prestam esse tipo de serviço já oferecem a possibilidade de configuração de algumas dessas variáveis, tornando a infra-estrutura mais moldável para atender as especificações do cliente. A idéia de ter *software* como serviço resolve alguns problemas de utilização, por exemplo, as licenças proprietárias. Torna-se dispensável a compra de *hardware* para suportar o *software* ou mesmo a compra de licença de outros *softwares* para suporte ao primeiro. A redução de custos com licenças proprietárias usando outras mais flexíveis como assinaturas ou *pay-as-you-go*, onde o usuário paga somente pelo tempo que usar do sistema, torna o modelo mais atrativo.

Segundo (ZHANG; CHENG; BOUTABA, 2010), esse novo paradigma oferece para empresas de todos os níveis algumas alternativas interessantes que tornam o seu uso atraente para o mercado. Algumas alternativas do recurso são:

- Não há necessidade de um investimento inicial grande: nuvens computacionais que oferecem contratos da forma *pay-as-you-go* dão ao usuário a possibilidade de iniciar suas tarefas sem ter que fazer um investimento em máquinas físicas e *softwares*. Além de possibilitar o arrendamento dos recursos que ele efetivamente irá usar e pagar somente por isso.

- Infra-estrutura sob demanda: os recursos da nuvem computacional são alocados e desalocados de acordo com as necessidades do cliente. Então, o mesmo, não precisa pagar por uma infra-estrutura que atenda ao seu pico de utilização. Estes custos de operação são reduzidos quando não se exige o máximo poder computacional.

- Escalabilidade: a nuvem computacional nos dá uma noção de ter poder computacional infinito. A abstração do usuário quanto ao escalonamento dos recursos para atender a todos os processos oferece a ele a sensação de que a infra-estrutura não é mais uma barreira. Sempre que precisar de mais poder computacional a nuvem computacional, sendo um serviço, estará pronta para provê-lo.

Podemos ver que a computação em nuvens oferece grandes facilidades e retira, num primeiro momento, as barreiras impostas por limitações e configurações de infra-estrutura. Contudo, para oferecer as facilidades de uma infra-estrutura de *hardware* e *software* praticamente infinita e totalmente transparente ao usuário é necessário um árduo trabalho

de monitoramento de recursos de *hardware* e, em alguns casos, um gerenciamento de aplicações para oferecer o desempenho que o usuário espera. Também é de fundamental importância manter a qualidade do serviço (QoS) que engloba o funcionamento da nuvem computacional, a disponibilidade em tempo integral do serviço oferecido, a integridade dos dados e do processamento e a garantia de segurança dos dados e do processamento.

A figura 3.2 demonstra as diversas formas de acesso, usando celulares e computadores, aos serviços de uma nuvem computacional utilizando a internet como meio de comunicação. Também mostra que um provedor de serviços pode oferecer os diferentes tipos de computação em nuvens.

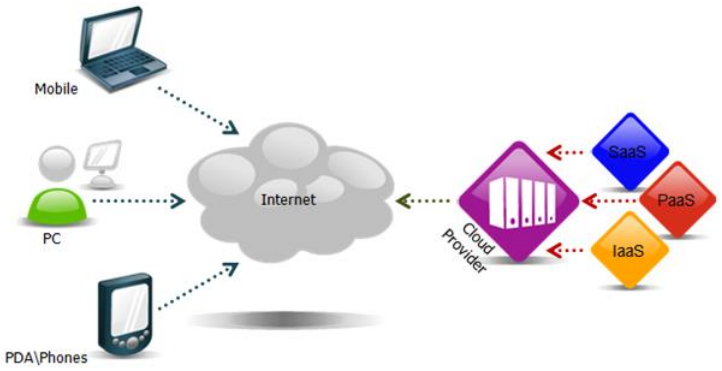

Figura 3.2: Meios de acesso a computação em nuvens. Fonte: http://contactdubai.com/tag/utility-computing - 06/2010

Nos próximos capítulos serão descritos os problemas de implementação de uma infra-estrutura de computação em nuvens e algumas soluções para contornar as dificuldades. Iremos mostrar também as soluções propostas no protótipo SGCCloud, que inicialmente será um serviço que oferece desktops virtuais para uso nos projetos do laboratório SGCLab, mas que posteriormente deverá se tornar um portal completo disponibilizando vários recursos para aproveitamento da infra-estrutura do SGCCloud.

3.1 Computação em nuvens, uma nova idéia?

Durante este capítulo será analisada as propostas do paradigma de computação em nuvens, mostrando que as propostas desse modelo já haviam sido discutidas em outras propostas anteriores.

"O interessante sobre computação em nuvens é que redefinimos computação em nuvens

para incluir tudo que fazíamos. (...) Eu não entendo o que temos que fazer de diferente sobre computação em nuvens, além de mudar a redação de nossos anúncios."
Larry Ellison, Wall Street Journal, 26 de setembro de 2008

Ao contrário do que parece, a computação em nuvem está longe de ser uma idéia revolucionária que veio para fazer uma mudança de paradigma completa. Todo o conceito é baseado em idéias que já existiam desde a década de 60. J.C.R. Licklider, um dos líderes do grupo *Advanced Research Projects Agency* (ARPA), que se destinava a pesquisas militares para o uso de computadores em comunicações, foi responsável em 1969 pelo projeto ARPANET *(Advanced Research Projects Agency Network)*, precursor da internet atual. Sua visão era que todas as pessoas, em qualquer lugar do mundo, tivessem a oportunidade de acessar informações e *softwares* disponíveis em qualquer outro ponto do planeta. A idéia de computação como serviço e de poder computacional sob demanda vem sendo fonte de inspiração para os pesquisadores desde o surgimento e da popularização da internet.

"Se os computadores que eu já defendi se tornarem os computadores do futuro, então a computação pode um dia ser organizada como um serviço público tal como o sistema de telefonia ou outro serviço público. (...) O computador poderia se tornar a base de uma indústria nova e importante."
John McCarthy, MIT Centennial, junho de 1961

John McCarthy é considerado o primeiro a sugerir que um dia a computação poderia ser comercializada em um modelo de negócios de serviço público como eletricidade e água. Durante a década de 60 essa idéia, e várias outras que serviriam de base para criação de um serviço de computação pública como computação distribuída e virtualização, eram muito populares, mas no inicio da década de 70 foram perdendo a força devido a limitações tecnológicas da época.

Por volta do ano 2000, essas idéias voltaram a ganhar força devido ao surgimento das ferramentas que faltaram nos anos 60 para implantação dos conceitos de computação pública e da evolução dos dispositivos de *hardware*. Podemos citar como exemplos a *WEB 2.0*, a virtualização que voltou em 1999 para solucionar os problemas de subutilização de servidores de aplicação e as experiências e estudos feitos com *clusters* e grades computacionais que ofereceram uma infra-estrutura capaz de fornecer poder computacional sob demanda para o usuário de forma a aperfeiçoar a utilização das máquinas.

3.2 Dificuldades a serem enfrentadas

Apesar das vantagens, a computação em nuvens talvez não seja uma solução para ser adotada em todos os casos. Ainda não é bem aceito a idéia de dados disponíveis publicamente e essa solução ainda é considerada inviável pelas empresas. Fatores que causam desconfiança no mercado são o fato dela ser considerada uma tecnologia pouco madura e o fato de ser um termo ainda muito ambíguo, através do qual as empresas não compreendem seu funcionamento. Outros teóricos mais radicais desacreditam totalmente no sucesso e afirmam que a computação em nuvens nunca terá os requisitos necessários para atender as corporações.

Uma das grandes dificuldades da computação em nuvens é migrar os sistemas que existem em *datacenters* para seu ambiente. Todos os principais provedores de computação em nuvens impuseram uma arquitetura diferente das utilizadas atualmente nas empresas. Mesmo com todas as vantagens oferecidas pelas provedoras, há um grande desânimo quando as pessoas se deparam com as dificuldades de migração dos sistemas e com a mudança de paradigma, onde agora elas não têm mais que monitorar *hardware* e sim *software*.

Alguns problemas no mesmo âmbito é a migração entre as arquiteturas de computação em nuvens. Isso se dá pela falta de um padrão de projetos para aplicações desenvolvidas pelas diferentes empresas. Isso causa grande desânimo quando as pessoas se deparam com as dificuldades que a migração pode propiciar.

Como todas as grandes mudanças que já aconteceram no decorrer da história, essa também traz resistência. Algumas alternativas que podem ajudar nesse processo são as definições de um padrão comum entre as plataformas de computação em nuvem, uma metodologia ou ferramenta para auxiliar e diminuir os custos da migração das aplicações em *datacenters* para a plataforma de uma nuvem computacional e a definição de uma SLA *(service level agreement)* para garantir a qualidade e disponibilidade do serviço da nuvem computacional, trazendo um pouco mais de confiança ao serviço prestado.

Capítulo 4

Virtualização

A virtualização é uma técnica que permite a criação de ambientes que simulam sistemas operacionais completos compartilhando recursos de *hardware* em uma máquina, estes recursos são desde tempo de processamento, uso da memória, acesso a disco e até dispositivos de I/O.

O assunto surgiu no início dos anos 70 (CARISSIMI, 2008), com a motivação de compartilhar o uso dos recursos dos caros e complexos mainframes da época. Foi utilizada com sucesso pela IBM na linha de *mainframes* IBM/370 e posteriores, onde oferecia máquinas virtuais onde rodavam suas aplicações.

Hoje a busca é por uma eficiência energética e por um melhor uso dos recursos de hardware adquiridos por corporações. As empresas procuram reduzir custos com *hardware* ao mesmo tempo adquirindo novos servidores, que consomem menos, são mais eficientes quanto ao poder de processamento (através de processadores *multicore*), gerando uma busca por tecnologias que permitam rodar múltiplos serviços sobre um único servidor ou conjunto deles.

A adoção da virtualização propiciou o uso compartilhado deste poder de processamento, onde os serviços destes servidores são virtualizados, compartilhando seus recursos e ao mesmo tempo facilitando o seu gerenciamento.

Outras empresas encontraram nesta metodologia, uma forma de recuperar seus gastos com *hardware*, onde disponibiliza recursos ociosos para empresas pequenas e usuários que não tenham, ou não pretendam investir em novos dispositivos, em troca de assinaturas e cobranças por uso.

Estas nuvens computacionais fornecem uma forma das empresas virtualizarem seus

serviços remotamente, mas sem que o usuário tenha conhecimento de onde os estão executando-os. Utilizando-se de virtualização, criam máquinas virtuais que terão seus serviços rodando sobre uma infra-estrutura física compartilhada de grande poder computacional.

Entre os serviços mais utilizados são os *storages*, serviços críticos que necessitam de grande carga computacional e alta disponibilidade, como servidores *web*, *backup*, entre outros.

4.1 Conceitos envolvidos

Para explicar melhor a essência da virtualização, temos antes que introduzir alguns conceitos básicos.

As máquinas virtuais (MV) são ambientes independentes de seus hospedeiros, isto é, possuem seu próprio sistema operacional e demais aplicativos e serviços, sem que estes afetem o funcionamento do sistema que as hospedam. Podem comunicar entre si, com o hospedeiro e com outras máquinas através de uma rede. Para acesso ao *hardware* físico, utiliza *drivers* que simulam a existência do dispositivo para a máquina virtual e que realizam a comunicação com o verdadeiro.

Um hospedeiro ou *host* é a máquina física onde rodam estas máquinas virtuais. Ele normalmente possui *softwares* para gerenciar a comunicação entre o *hardware* e elas, denominado monitor de máquinas virtuais (MMV) ou *virtual machine monitor* (VMM).

Quando este monitor trabalha em uma camada dentro do sistema operacional "ligando"diretamente o *hardware* às máquinas virtuais são chamados *hypervisors*. Eles podem estar ligados ao sistema operacional do hospedeiro, através de uma modificação do *kernel*, ou ser um sistema independente com *kernel* próprio.

4.2 Tipos de Virtualização

A implementação de máquinas virtuais podem ser obtidas através de duas técnicas principais: virtualização total ou completa e paravirtualização, que serão abordadas nos próximos tópicos.

Existem também outras formas de virtualização, como a de *hardware* e de aplica-

tivos. A virtualização de *hardware* consiste em simular a existência de um dispositivo que, na verdade, não existe fisicamente na máquina. Esse tipo de virtualização é usado principalmente com placas de rede e dispositivos de I/O.

A virtualização de aplicativos é quando uma aplicação é executada utilizando o auxílio de uma máquina virtual que interpreta o código e o executa no sistema operacional do hospedeiro. Um exemplo é a linguagem Java que permite que um programa execute da mesma forma, independente do sistema onde roda, pois sua máquina virtual *(Java virtual machine)* interpreta os *bytecodes* gerados pelo compilador Java e os transforma em instruções de máquina para execução no sistema.

4.2.1 Virtualização Total

A virtualização total (GONÇALVES; JUNIOR, 2010) é uma técnica que recria toda a infra-estrutura de *hardware* existente em uma máquina hospedeira para que as máquinas virtuais executem como tivessem seu próprio *hardware*.

Utiliza o monitor de máquinas virtuais para o gerenciamento e comunicação das máquinas virtuais com o *hardware* hospedeiro. Também permite que o sistema operacional usado seja diferente entre o hospedeiro e o hóspede, sem que haja modificações nos seus respectivos *kernels* para seu funcionamento.

Possui algumas desvantagens: as instruções executadas na máquina virtual são interpretadas pelo monitor de máquinas virtuais e este define se as mesmas são sensíveis ou não, antes de executá-las realmente, o que gera um custo de processamento a mais para isso. Uma instrução sensível é uma instrução que, executada na máquina virtual, pode afetar o funcionamento de aplicações da máquina hospedeira, como por exemplo, instruções que alocam espaços em memória para uso da máquina virtual e que estejam sendo usados por uma aplicação na máquina hospedeira, afetando sua execução e podendo gerar falhas. A troca de um item de *hardware* do hospedeiro também pode ocasionar problemas se o mesmo não tiver suporte no VMM.

Na figura 4.1, mostra como são as camadas de um hospedeiro que usa virtualização total. Mais abaixo o *hardware* físico, depois o sistema do hospedeiro e o monitor de máquinas virtuais, uma aplicação que gerencia a comunicação e execução das máquinas virtuais. Acima tem as máquinas virtuais com seus próprios itens, isoladas umas das outras e independentes do sistema hospedeiro. Os *drivers* são *hardwares* simulados para que a máquina virtual possa se comunicar com os *hardwares* físicos. O sistema operacional

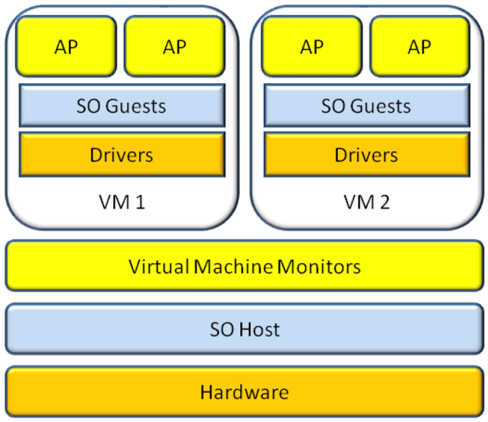

Figura 4.1: Virtualização Total.

da máquina virtual, que é independente do hospedeiro e suas próprias aplicações.

4.2.2 Paravirtualização

Segundo (CARISSIMI, 2008), é uma abordagem alternativa para contornar as desvantagens da virtualização total. O sistema hóspede é modificado para realizar chamadas para o monitor de máquinas virtuais sempre que uma instrução é considerada sensível, não sendo mais necessários os testes de instrução.

Máquinas virtuais modificadas acessam ao *hardware* através dos *drivers* da máquina hospedeira, o que aumenta o desempenho do uso dos recursos computacionais do hospedeiro.

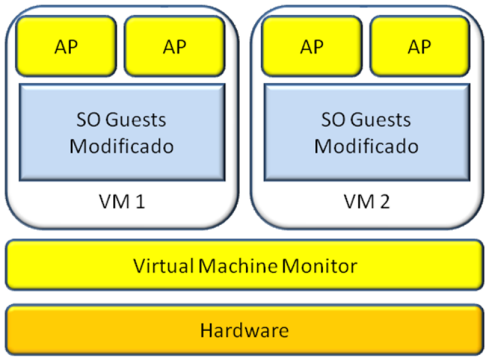

Figura 4.2: Paravirtualização.

Na figura 4.2, a estrutura de um servidor usando paravirtualização. Em cima da camada de *hardware* do hospedeiro, existe a camada do monitor de máquinas virtuais, que neste caso pode ser uma modificação no *kernel* do sistema do hospedeiro ou um sistema próprio para controlar as máquinas virtuais. Como tem acesso direto ao *hardware* do

sistema base, as máquinas virtuais não possuem uma camada de *hardware*. Seus sistemas operacionais recebem modificações para se comunicar com o *hardware* através do monitor de máquinas virtuais, aqui denominado *hypervisor*.

Comparando virtualização total e paravirtualização temos que com a virtualização total, uma máquina virtual não necessita de modificações em seu sistema operacional para ser executado em um sistema hospedeiro, ao passo que a paravirtualização exige que o *kernel* de sua máquina virtual seja modificado para se comunicar com o *hypervisor*. Em contrapartida, a paravirtualização explora de melhor maneira os recursos computacionais do sistema hospedeiro, por ter acesso ao direto *hardware* real da máquina física, obtendo um melhor desempenho do que a virtualização total. Entretanto, esta diferença de desempenho vem diminuindo com o auxílio de dispositivos de *hardware* com suporte a virtualização presentes em processadores desenvolvidos por empresas como a AMD e a Intel.

4.3 Propriedades de Virtualização

Algumas propriedades presentes em virtualização, importantes por garantir que máquinas virtuais funcionem independentemente umas das outras e do sistema hospedeiro, sem afetá-lo:

- Isolamento: processos executados em uma máquina virtual não devem interferir na execução de outras máquinas virtuais e no hospedeiro.
- Inspeção: o VMM tem controle total e acesso as informações das execuções de processos em suas máquinas virtuais.
- Interposição: o VMM deve ser capaz de inserir instruções na máquina virtual, como por exemplo, instruções que transfiram arquivos da máquina virtual para a máquina física e vice-versa.
- Eficiência: instruções consideradas não sensíveis devem ser executadas diretamente sobre o *hardware*, como instruções de acesso a dispositivos de I/O.
- Gerenciabilidade: o VMM deve gerenciar uma máquina virtual independente da existência de outras máquinas virtuais.

- Compatibilidade de software: todo software escrito para uma determinada plataforma deve ter sua execução em uma máquina virtual idêntica a sua execução em uma máquina física desta mesma plataforma.

4.4 Vantagens da virtualização

Segundo (MENASCÉ, 2005), estas são as principais vantagens da virtualização:

- Segurança: o isolamento das máquinas virtuais permite que cada uma tenha seu próprio sistema operacional, apropriado para os serviços que rodarão nelas. Em caso de uma máquina virtual seja invadida, não comprometerá as demais, devido a este isolamento.
- Disponibilidade e confiabilidade: uma falha de *software* ocorrida em uma máquina virtual não deve afetar outras máquinas virtuais.
- Custo: a possibilidade de redução de custos pela consolidação de servidores menores em servidores mais poderosos. Esta economia em *hardware* gera redução de custos com operação, como espaço no local, pessoal para manutenção e licenciamento de *software*.
- Adaptabilidade a variação do trabalho: mudanças intensas na carga de trabalho e falhas de recursos físicos são contornadas com a troca de recursos entre máquinas virtuais.
- Balanceamento de carga: devido ao encapsulamento de um *software* dentro de uma máquina virtual, permite que um usuário administrador use o VMM para migrar esta máquina virtual para outra plataforma a fim de melhorar o desempenho.
- Aplicações Legadas: se uma empresa decide migrar de plataforma, pode manter máquinas virtuais com o sistema operacional anterior a fim de manter a execução de aplicações legadas e reduzir custos de migração.

4.5 Exemplos de softwares de virtualização

Existem empresas desenvolvendo soluções em virtualização para o mercado corporativo, entre as quais as principais utilizadas são:

4.5.1 VMware

Companhia considerada líder do mercado de virtualização (VMWARE, 2010). Fundada em 1998, trouxe de volta a virtualização para os dias de hoje, onde empresas gastam muitos recursos para manutenção de seus parques de máquinas. Cresceu oferecendo soluções que simplificam a infra-estrutura de TI das empresas, gerando economia de energia, pessoal e financeiro, permitindo para que os investimentos fossem direcionados para a eficiência destes parques.

Possuem soluções que abrangem desde virtualização de *desktops*, virtualização de servidores e mais recentemente o mercado de computação em nuvens. Adotando a computação verde como lema, fornece infra-estruturas virtuais capazes de alocação de recursos entre máquinas virtuais, migração, *backup*, serviços de *storage* entre outros.

Seus produtos utilizam como base a virtualização total, mas algumas aplicações também dão suporte a paravirtualização, através do suporte a virtualização em *hardware* presentes nos processadores mais recentes.

Estre seus produtos mais utilizados estão:

- VMware Server: aplicação gratuita para virtualização de servidores. Funciona como um *software* instalado no servidor físico, compartilhando os recursos com diversas máquinas virtuais. Permite a criação e gerenciamento de diversas máquinas virtuais.
- VMware Player: destinada a *desktops*, permite que usuários possam executar e experimentar outros sistemas operacionais.

4.5.2 Xen

Surgiu nos anos 90 (BARHAM et al., 2003) como um projeto de pesquisa da Universidade de Cambridge. Criaram um *hypervisor* para o projeto Xenoserver.

Posteriormente fundaram a XenSource para transformar o *hypervisor* em uma ferramenta de pesquisa para o mercado, como uma solução *open source* com os conceitos de paravirtualização. Foi comprada pela Cytrix Systems no ano de 2007.

Hoje, eles desenvolvem uma plataforma de computação em nuvens chamada *Xen Cloud Platform*.

Desenvolvido sobre os conceitos de paravirtualização, o Xen é uma plataforma que

oferece diversos serviços, como migração de máquinas virtuais entre máquinas hospedeiras, onde uma máquina virtual troca de *host* enquanto executa. Gerência de multiplos hospedeiros, *storage* centralizado entre outros serviços ligados a computação em nuvem.

4.5.3 VMware x Xen

Avaliando as duas soluções verifica-se vantagens e desvantagens entre ambas as tecnologias. O VMware, usando virtualização total, permite que as máquinas virtuais tenham uma variedade maior de sistemas operacionais compatíveis já que não necessita de modificações no *kernel*. O Xen, por outro lado, usa paravirtualização e fornece um suporte limitado de sistemas que podem torna-se máquinas virtuais, pois as mesmas precisam de *kernels* modificados para sua comunicação com o *hypervisor*, mas devido esta modificação, otimiza o uso do *hardware* da máquina física, tendo um desempenho melhor.

Quanto a facilidades de uso, o VMware funciona como uma aplicação de gerenciamento de máquinas virtuais com uma interface amigável para o usuário, enquanto o Xen tem uma interface mais técnica que exige um pouco mais de conhecimento sobre gerência de sistemas. Atualmente oferecem quase os mesmos serviços, como migração de máquinas virtuais, armazenamento centralizado, controle corporativo de redes e infra-estrutura, entre outros.

Capítulo 5

SGCCLoud

Antes de apresentar o portal SGCCloud, falaremos sobre alguns projetos e trabalhos relacionados a computação em nuvens. São projetos desenvolvidos internamente no laboratório do SGCLab e projetos externos considerados interessantes e que forneceram idéias para a implementação e trabalhos futuros.

5.1 Trabalhos Relacionados

5.1.1 EasyGrid AMS

EasyGrid AMS é um *middleware* para execução de aplicações MPI em ambientes de grades computacionais. Ele encapsula a aplicação desenvolvida para execução em ambientes de *clusters* e realiza o gerenciamento para que a aplicação execute de forma eficiente em uma grade computacional.

Ele funciona de forma a garantir que a aplicação se adapte de forma eficiente ao ambiente e a suas constantes mudanças. Os processos são capazes de adotar diferentes políticas de escalonamento dinâmico e de tolerância a falhas de acordo com suas necessidades para aproveitar da melhor forma os recursos da infra-estrutura (RODRIGUES, 2009).

O EasyGrid AMS trabalha com aplicações do tipo *Bag of Tasks* e DAG (*Directed Acyclic Graph*). Em relação ao escalonamento, o middleware usa uma solução hibrida, a aplicação se inicia com um escalonamento estático, mas depois podem ser realocadas para melhor adaptação ao ambiente (NASCIMENTO et al., 2008). Possuem também um sistema de tolerância a falhas que permite a recuperação em caso de falha de algum recurso sem

a necessidade da interrupção da execução da tarefa (SILVA, 2010).

Apesar de ser desenvolvido para ambiente de grades computacionais, o EasyGrid AMS pode ser utilizado em ambientes de computação em nuvens, pois como já foi discutido nos capítulos anteriores a organização das máquinas nessas duas abordagens são bem parecidas. O EasyGrid AMS ainda não foi incorporado na versão atual do SGCCloud, mas será a peça chave para atender aos requisitos da IaaS. Para tal devemos primeiramente realizar testes com o EasyGrid em máquinas virtuais interligadas e observar seu comportamento e eventuais perdas de desempenho, contudo em uma observação inicial não parece haver empecilhos para o funcionamento da mesma em um ambiente totalmente virtual.

5.1.2 MyEasyGrid Portal

MyEasyGrid Portal (VENITO, 2006), visa facilitar o uso de grades computacionais com a utilização de uma interface para seu acesso e utilização dos recursos da grade. É um projeto totalmente baseado na *web* que traz as vantagens de requisitos mínimos para acesso e apenas conhecimento de acesso a páginas *web* por parte de seus usuários.

Sua proposta é oferecer o acesso a grade a usuários do meio acadêmico sem que os mesmos precisem ter um conhecimento sobre a manipulação de uma infra-estrutura baseada em máquinas heterogêneas. Para tal ele oferece uma interface amigável através da tecnologia de *portlets* que permite que o usuário possa submeter suas aplicações à execução em um ambiente de grades.

Para operar as máquinas da grade o MyEasyGrid Portal se comunica com elas através do *middleware Globus Toolkit* (FOSTER; KESSELMAN, 1996) que é uma referência na comunidade científica de computação de alto desempenho para a implantação de grades computacionais.

Em muitos aspectos, foram utilizadas as idéias do MyEasyGrid portal para implementação do portal do SGCCloud, como uma interface para o usuário acessar a nuvem computacional e ter acesso a grade virtual do usuário, e assim executar suas aplicações.

5.1.3 OpenNebula

É um *middleware* de gerenciamento para a computação em nuvens que oferece fácil manipulação dos recursos físicos e virtuais (OPENNEBULA, 2010). Esse projeto foi iniciado pensando em oferecer uma opção a empresas e projetos que precisem de grande poder

computacional a montar e gerenciar suas infra-estruturas de forma simples.

Ele oferece todo o controle de infra-estrutura necessária (virtualização, armazenamento e comunicação), alocação dinâmica de recursos e promete uma integração simples entre uma nuvem computacional proprietária e pública (nuvem computacional híbrida) para facilitar o aumento de recursos o quando for necessário.

Outro exemplo deste tipo de software é o Eucalyptus (EUCALYPTUS, 2010), um software open-source para implementação de IaaS baseadas em uma grande variedade de sistemas Linux e suportando diversas ferramentas de virtualização para implantação de computação em nuvens.

5.1.4 Cloud Views

O Cloud Views é uma proposta para implementação de um *storage* em cima de nuvens computacionais públicas ou privadas (GEAMBASU; GRIBBLE; LEVY, 2009). A idéia é não somente armazenar quantidades enormes de dados, mas também compartilhar esses dados de acordo com *views*. Cada *view* dá acesso a um conjunto específico de dados e para um usuário é dada a permissão de acesso a uma *view*.

A proteção dos dados é feita através de *views* assinadas, onde a própria *view* é responsável por verificar as assinaturas e dar acesso somente a pessoas com certificados válidos para acessá-las.

5.1.5 Trusted Cloud Computing Platform (TCCP)

A proposta do TCCP é dar condições a IaaS de garantir de confidencialidade na execução de uma tarefa. O usuário tem meios de atestar se o ambiente de execução está seguro para seu uso (SANTOS; GUMMADI; RODRIGUES, 2009).

Para garantir a integridade da infra-estrutura o sistema conta com duas ferramentas:

- *Trusted Virtual Machine Monitor* (TVMM): monitora as máquinas virtuais e da acesso a inspeção e modificação somente a usuários autorizados.
- *Trusted coordinator* (TC): monitora todos os nós e atesta se eles são nós confiáveis ou não. Ele considera um nó confiável se o nó está dentro dos parâmetros de configuração especificadas e que esteja rodando um TVMM confiável.

5.2 O Portal

Um portal *web* são páginas na internet que servem de ponto de acesso a serviços e informações. Estes serviços podem ser aplicações ou acesso a recursos remotamente. O SGCCloud é um portal de acesso e gerenciamento a infra-estrutura de computação em nuvens, oferecendo os serviços de uma nuvem computacional do tipo *Infrastructure as a Service* (IaaS).

O portal permite que um usuário selecione a partir de uma interface, máquinas virtuais para montar sua grade virtual pessoal. Se o usuário tivesse que fazer isso manualmente, ele deveria ter conhecimentos sobre o sistema de virtualização utilizado e ter acesso aos servidores das máquinas virtuais para que pudesse montar sua grade de máquinas virtuais.

Uma grade virtual é composta por máquinas virtuais que podem estar em um mesmo servidor ou em servidores diferentes desde que as mesmas possam se comunicar. Estas máquinas virtuais possuem um servidor de arquivos em comum para que o usuário envie seu arquivo e compartilhe entre todas as máquinas virtuais e possa rodar suas aplicações paralelas, recuperarando o resultado depois.

A grade virtual é usada pelo usuário para executar suas aplicações paralelas de forma isolada, sem que aplicações de outros usuários interfiram nas suas, pois a grade de um usuário é feita com máquinas virtuais diferentes das usadas por outros usuários do portal. Após o uso da grade virtual, o usuário retorna ao portal para liberar os recursos para que outros possam usá-los.

A infra-estrutura de computação em nuvens é composta por um servidor onde fica o portal e por máquinas denominadas clientes, que são servidores para as máquinas virtuais que compõem os recursos disponíveis para a nuvem computacional. Estes clientes podem ser organizados em *sites* conforme sua distribuição física.

O portal cadastra estes *sites* e toda sua infra-estrutura, suas máquinas e máquinas virtuais em um banco de dados para serem acessados conforme a necessidade e o tipo de usuário. Foram definidos três tipos de usuários:

- Admin: um usuário *admin* é o gerente da infra-estrutura, ele tem permissão para adicionar usuários e definir seus privilégios, criar *sites* e definir administradores dos mesmos, assim como adicionar máquinas e máquinas virtuais aos *sites*. Também é um usuário do portal que pode selecionar máquinas virtuais e montar uma grade virtual.

- Admin Site: um usuário que administra os computadores clientes de um *site* junto a nuvem computacional. Ele pode adicionar novas máquinas e máquinas virtuais e também usar o portal como um usuário comum.

- Comum: um usuário comum tem apenas acesso as máquinas virtuais disponíveis para a criação de uma grade virtual particular. Ele seleciona quantas máquinas virtuais precisa para rodar sua aplicação e as inicializa. Ao término do uso, ele as libera para que outros usuários utilizem.

Ao selecionar máquinas virtuais para uma grade virtual, um usuário não tem a noção de onde elas estão rodando, apenas que estão dentro da nuvem computacional. Estas máquinas virtuais podem estar em um mesmo servidor compartilhando recursos ou em servidores diferentes dependendo de quais forem selecionadas pelo usuário.

A figura 5.1 exemplifica o acesso de um usuário a infra-estrutura da nuvem computacional. Ele solicita as máquinas virtuais ao servidor; este verifica em que recursos elas estão cadastradas e as inicializa e posteriormente libera o acesso ao usuário. No exemplo o cliente um solicita três máquinas virtuais para sua grade virtual e o cliente dois apenas uma que pode funcionar como *desktop* remoto para ele.

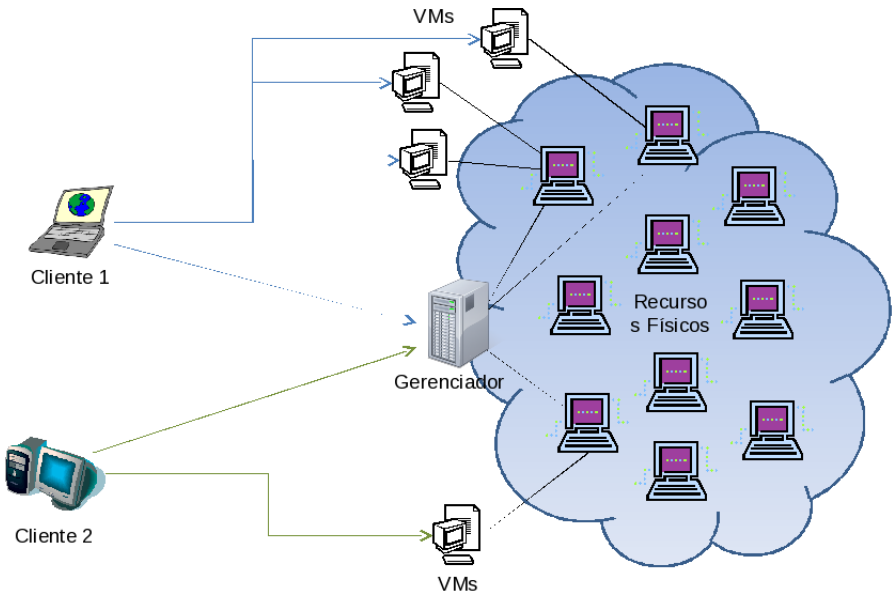

Figura 5.1: Modelo de acesso de usuários a nuvem computacional.

Foi desenvolvido com a linguagem Java e o pacote de desenvolvimento web GWT (SMEETS; BONESS; BANKRAS, 2009). O GWT foi criado pelo Google para possibilitar a criação de um *front-end* Ajax com Java e o compila para JavaScript. É um pacote de

fácil utilização e reusabilidade que permite fazer alterações na interface sem precisa de recompilação, apenas atualizando o navegador e revendo a página.

Para a comunicação entre o usuário e o servidor, usa-se o GWT RPC, ele permite chamar métodos remotamente. Ele serializa os argumentos e invoca um método correspondente no servidor e desserializa o valor de retorno para o usuário.

Para esse projeto foi utilizado um banco de dados MySQL para armazenamento de todas as informações relacionadas a infra-estrutura da nuvem computacional, porém qualquer outro gerenciador de banco de dados poderia ter sido utilizado com algumas mudanças de parâmetros e do *driver* correspondente.

5.2.1 Cadastro e identificação de um usuário

Inicialmente o usuário precisa ser cadastrado por um usuário *admin* através de uma página que contém as informações pessoais para identificação do mesmo e definindo seu perfil conforme a figura 5.2.

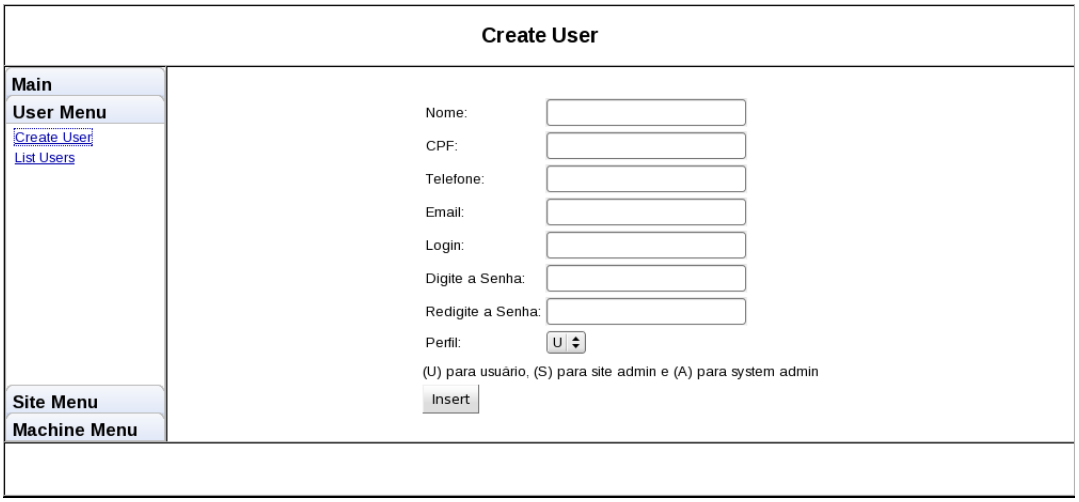

Figura 5.2: Tela de criação de usuários.

Posteriormente para entrar no portal, o usuário devidamente cadastrado é identificado através de uma tela de *login*. A tela verificará se o *login* e a senha estão no banco de usuários e confere como na figura 5.3:

5.2.2 A visualização da infra-estrutura da nuvem computacional

Após se logar, o usuário tem opções diversas, conforme seu perfil. A principal delas

Figura 5.3: Tela de Login.

e comum a todos é a de visualizar a infra-estrutura da nuvem computacional. Um *admin* consegue ver todos os *sites* e suas respectivas máquinas e máquinas virtuais para utilizá-las. Um *site admin*, consegue ver apenas as máquinas dos *sites* que administra e as demais máquinas virtuais. Um usuário comum consegue ver apenas as máquinas virtuais. Na figura 5.4 temos uma tela com a estrutura em árvore que tem o *site* como raiz, as máquinas cliente como ramos e as máquinas virtuais como folhas.

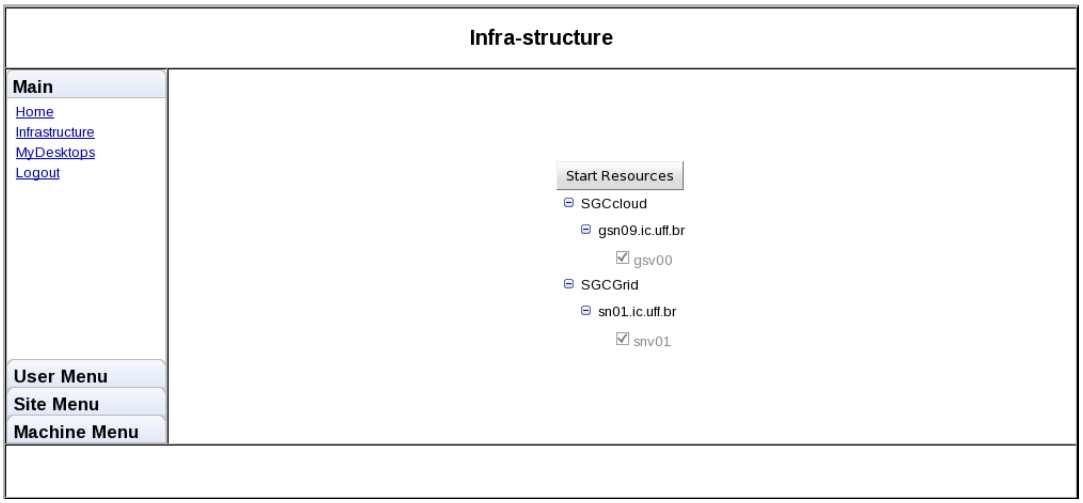

Figura 5.4: Tela com a Infra-estrutura.

Nesta tela os usuários podem selecionar as máquinas virtuais que pretende utilizar e montar sua grade virtual. Posteriormente poderá enviar seus arquivos para um servidor NFS que interligará os discos das máquinas virtuais.

5.2.3 Visualização das informações de um site

Um *site* é cadastrado apenas com um nome, que deve ser único no banco. A ele vinculamos os usuários com privilégios de *site admin* para administrá-lo, ou seja anexar suas máquinas e máquinas virtuais. A figura 5.5 mostra um *site* e sua lista de administradores.

Figura 5.5: Tela de visualização de um site.

5.2.4 Cadastro das máquinas clientes

Uma máquina cliente é cadastrada a partir de um *site* e indicando seu *hostname* e IP e seu estado para uso, este estado indica se ela está disponível para a nuvem computacional ou foi apenas cadastrada. Os demais dados pertinentes a máquina são captados pelo *middleware* e inseridos no banco. Usuários *admin* podem cadastrar quaisquer máquinas clientes em qualquer *site*, enquanto um *site admin* pode criar máquinas somente nos *sites* que administra.

A figura 5.6 mostra a tela de criação de máquinas clientes:

A figura 5.7 mostra os dados captados pelo *middleware* junto com os dados cadastrados:

5.2.5 Cadastro de máquinas virtuais

Uma máquina virtual é vinculada a máquina cliente onde rodará. É cadastrada com informações básicas para uso como seu nome de identificação no VMware, seu *hostname* e IP local e o sistema operacional que roda.

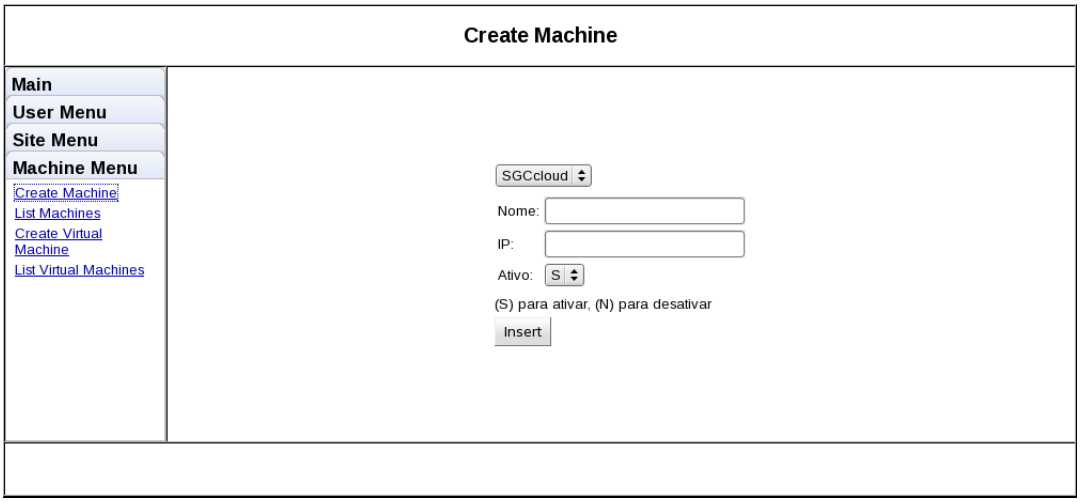

Figura 5.6: Tela de cadastro de máquinas.

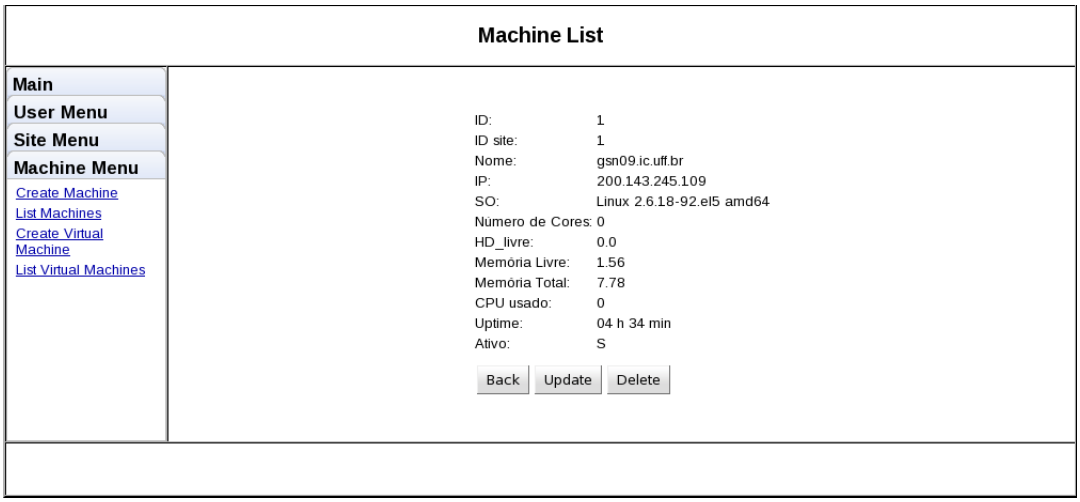

Figura 5.7: Tela de visualização de máquinas.

A figura 5.8 mostra a tela exibida para a criação de uma máquina virtual:

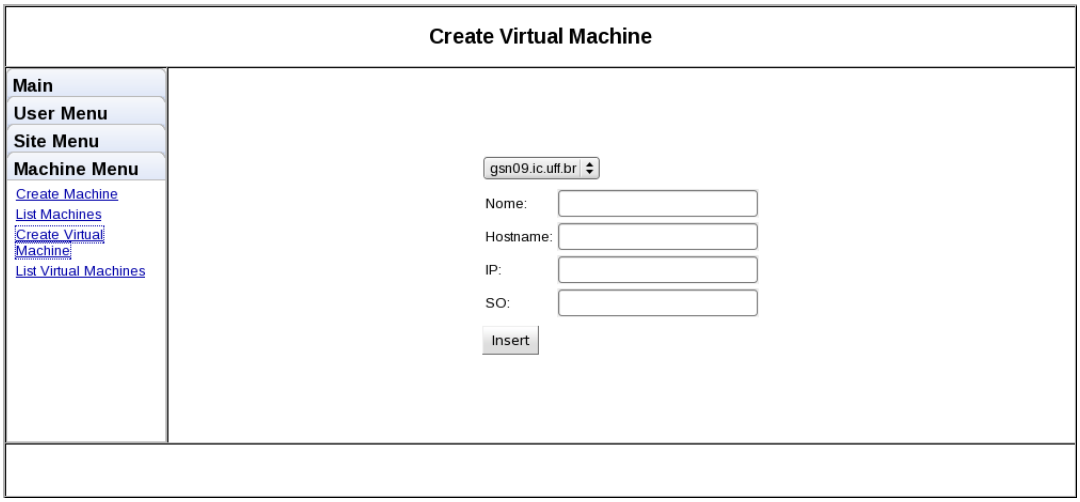

Figura 5.8: Tela de cadastro de máquinas virtuais.

Após sua criação podemos ver as informações que forem inseridas no banco. A figura 5.9 exibe esta saída.

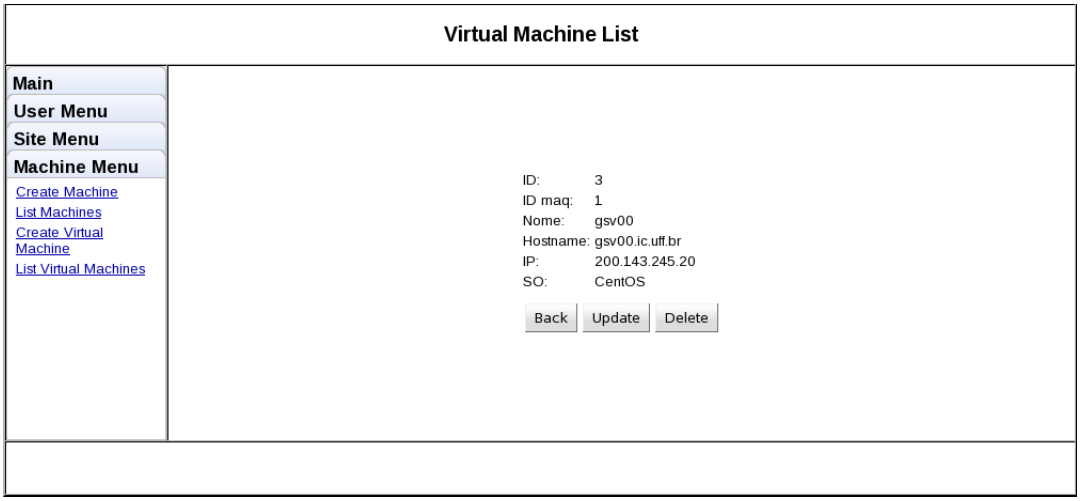

Figura 5.9: Tela de visualização de máquinas virtuais.

5.3 A infra-estrutura

A infra-estrutura é composta por um servidor centralizado que contém o portal e servidores NFS e NIS. Estes serviços serão usados posteriormente para ter acesso às máquinas virtuais através de um terminal Ajax a ser adicionado ao portal em uma atualização. O NFS é um serviço de compartilhamento de disco onde os usuários terão espaço para armazenar suas aplicações e executá-las em sua grade virtual. O NIS é um centralizador de contas que será sincronizado com o portal para que o usuário tenha acesso as suas máquinas usando o mesmo *login* e senha do portal.

O servidor também possuí um *middleware* para se comunicar com as máquinas clientes e inicializar as máquinas virtuais da nuvem computacional.

A infra-estrutura possuí também máquinas denominadas clientes, que podem ser servidores de grande porte ou um simples *desktop*. Para ser considera habilitada para executar máquinas virtuais, um cliente deve ter memória e processamento suficiente para suportar ao menos uma máquina virtual. Preferencialmente utiliza-se máquinas *multicore* com uma boa quantidade de memória RAM (acima de 2 GB).

O cliente terá um VMware Server instalado nele para gerenciar as máquinas virtuais localmente e um pacote VIX-API para que o *middleware* possa se comunicar através de linhas de comando injetadas no sistema hospedeiro. O pacote VIX é um conjunto de

instruções que rodam em um programa chamado *vmrun* que permite os mesmos controles das máquinas virtuais sem usar a interface do VMware Server.

A máquina virtual é criada no Vmware Server que possui um sistema operacional próprio e com algumas configurações de memória e rede pré-estabelecidas. Foram criadas para usarem até 1GB de memória e um *core* de processamento. Possuem sistema operacional Linux e tem seu próprio endereço IP. Posteriormente terão acesso a um servidor de contas NIS e ao servidor de disco NFS para acessar aos arquivos dos usuários que as utilizarão.

5.4 Middleware

O *middleware* é o componente responsável por fazer a comunicação entre o portal e a infra-estrutura. É nele que estão implementadas todas as funções de gerenciamento e interatividade com as máquinas físicas e virtuais. Nesse projeto, o *middleware* é totalmente desenvolvido em Java e é composto de um módulo servidor e módulos clientes (*daemon*) que por sua vez se comunicam através da API RMI.

A figura 5.10, representa como estão organizadas as máquinas clientes com relação ao servidor onde o portal está instalado, para a formação da infra-estrutura de computação em nuvens.

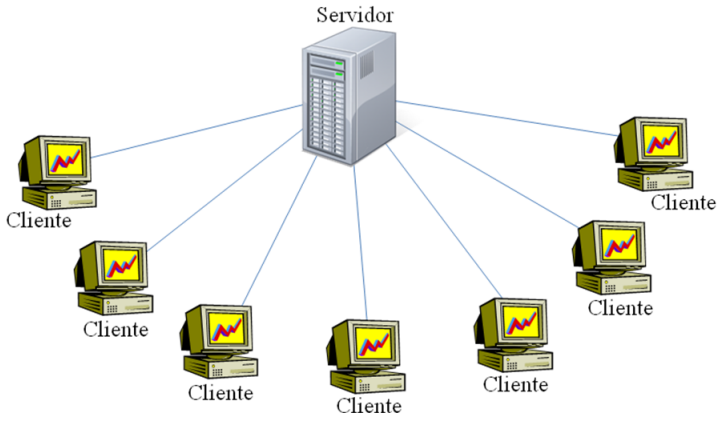

Figura 5.10: Exemplo da estrutura da nuvem computacional.

Adotamos a linguagem Java para o desenvolvimento do *middleware* por ser uma linguagem de amplo conhecimento da equipe de desenvolvimento, por ter fontes de pesquisa muito amplo e por ser uma linguagem portável. A independência de plataforma foi um fator importante na escolha do Java, devido à natureza heterogênea do modelo de compu-

tação em nuvens, o Java se aplica perfeitamente à proposta, pois é uma linguagem independente de plataforma. Com o "lema escreva uma vez, execute em qualquer lugar"(*write once, run anywhere*), máquinas diferentes que compõem a infra-estrutura podem executar o mesmo programa sem a necessidade de recompilação ou adaptação de código. A escolha também se ajustou bem com o portal, também desenvolvido em Java sob a API GWT, pois com os dois lados se comunicando na mesma linguagem, as informações podem ser transmitidas e entendidas sem a necessidade de tradução.

Contudo mesmo com todas as vantagens oferecidas pela linguagem, a mesma mostrou algumas grandes dificuldades. A falta de integração com o sistema operacional por motivos de segurança da linguagem causou problemas que foram impeditivos para implementação de um código único para plataformas Unix e Windows.

O funcionamento das camadas do *middleware* será descrito de forma mais datalhada.

5.5 Cliente

Como já foi dito, o cliente é um *daemon* desenvolvido em Java que fica ativo em *background* no sistema operacional, totalmente transparente e sem a necessidade de operação pelo usuário. O objetivo dele é efetivamente realizar as operações na máquina física de acordo com o pedido do servidor. As atividades do processo cliente são a coleta das diversas informações importantes para o sistema; comunicação com o VMware para operar as máquinas virtuais; receber e executar as requisições do servidor. Devido às facilidades proporcionadas pela camada de comunicação, qualquer outra atividade que tenha que ser implementada entre o servidor e o cliente poderá ser inserida facilmente implementando a ação na aplicação cliente e fazendo a chamada da mesma pelo servidor.

Durante a execução o impacto causado no sistema é imperceptível para o usuário. O mesmo não sente perda considerável de desempenho enquanto usa o recurso.

5.5.1 Coleta de informações

Uma das funções da aplicação cliente é coletar informações que são pertinentes para o conhecimento do sistema. Futuramente eventuais tomadas de decisão baseadas nessas informações poderão ser tomadas pelo servidor para maximizar a funcionalidade da nuvem computacional.

A tabela 5.1 lista todas as informações, uma descrição e a forma como ela está sendo obtida:

Informação	Detalhe	Forma de obtenção
Versão do SO	Uma especificação limitada sobre o Sistema Operacional. Ex.: Windows, Linux XXX.	System.getProperty("os.name") + System.getProperty("os.version") + System.getProperty("os.arch")
Processadores	Uma lista com informações sobre todos os processadores da máquina: • Modelo – modelo completo do processador; • Clock – clock do processador em Mhz; • Número de Cores – número de núcleos de processamento no processador;	Linux – Informações filtradas da saída do comando "grep processor |model name |cpu MHz /proc/cpuinfo" Windows – Não são colhidas essas informações.
CPU Usado	Porcentagem de CPU que está em uso no momento.	top -n 2 -b -d 0.2
Memória Livre	Quantidade de memória livre em kilobytes.	operatingSystemMXBean. getFreePhysicalMemorySize();
Memória Total	Quantidade de memória total em kilobytes.	operatingSystemMXBean. getTotalPhysicalMemorySize();
HD Livre	Espaço em disco livre em kilobytes.	File(<diretorio _raiz>). host.getFreeSpace();
Up Time	Quantidade de tempo que a máquina está disponível para a *cloud* em nanosegundos.	Uma contagem que se inicia quando o serviço do *daemon* se inicia.

Tabela 5.1: Tabela de Informações das máquinas.

Essas informações são divididas em dois grupos, as informações estáticas e as informações dinâmicas. Informações estáticas são as que não mudam sem que a máquina tenha que ser reiniciada. Essas informações são: versão do sistema operacional, informações dos processadores e memória total. As informações dinâmicas são monitoradas continuamente por sofrerem alterações durante a utilização do *hardware*: CPU usado, memória livre, HD livre e *uptime* (tempo que uma máquina cliente fica disponível para a nuvem computacional). Essas informações são coletadas com intervalo de tempo de 1 minuto.

Como podemos ver quase todas as informações são obtidas através da máquina virtual do Java. A vantagem dessa abordagem é manter a independência de plataformas e utilizar

a maior qualidade da linguagem que é a portabilidade entre as plataformas. Uma única implementação poderia funcionar da mesma forma em máquinas de arquiteturas e sistemas operacionais diferentes, tornando o código totalmente portável entre qualquer máquina, seja ela da plataforma Unix ou Windows.

Contudo esse objetivo não pode ser alcançado devido a algumas limitações da linguagem. Como já foi dito, devido à falta de interação da linguagem com o SO, as informações sobre o processador não puderam ser obtidas com uma abordagem puramente Java, tendo que ser obtida por chamadas específicas ao sistema Linux. Essa estratégia para obter informações comprometeu a portabilidade do código do processo cliente. Para gerar um *daemon* para a plataforma Windows essas chamadas Linux teriam que ser substituídas por chamadas ao sistema Windows. Outro problema é que algumas informações como tipo do sistema operacional são limitadas, não conseguimos saber qual a distribuição ou versão do sistema operacional. Esperamos futuramente conseguir uma abordagem totalmente nativa em Java, mas teremos que aguardar que alguma versão futura da *Java Virtual Machine* tenha uma integração maior com o sistema operacional.

Para resolver esse problema foi cogitada inicialmente a hipótese de escrever uma biblioteca JNI na linguagem C para obter essas informações, mas essa abordagem também não resolveria o problema da portabilidade, visto que a biblioteca teria que ser re-compilada em cada máquina que o *daemon* fosse instalado, então uma opção mais simples foi adotada.

Essas informações são um conjunto básico, qualquer nova informação que se tenha necessidade de monitorar deve apenas ser inserida no objeto que representa a máquina ou a máquina virtual e suas funções implementadas.

5.5.2 Interação com VMware

Cada máquina cliente tem instalado uma versão da aplicação VMware Server instalada juntamente com a api VIX, que permite acessos as máquinas virtuais através de linha de comando.

O *middleware* se comunica com o VMware usando esta API e um usuário cadastrado no VMware como administrador para acessar as máquinas virtuais, salvo em um arquivo criptografado gerado por um programa de configuração do *middleware* cliente. Entre esses acessos estão iniciar e parar uma máquina virtual e gerar um *snapshot* toda vez que a inicializar.

O *snapshot* garante que toda modificação feita durante a execução da máquina virtual possa ser desfeita e reveretida para seu estado inicial, pois salva o estado na sua inicialização. O estado da máquina virtual é composto por informações como o conteúdo que está no HD da máquina virtual, aplicações que estão em execução, espaços alocados em memória e no *swap*, entre outros.

5.6 Servidor

O módulo servidor é o responsável por operar todas as aplicações clientes. É através dela que as requisições são enviadas e os retornos das máquinas da infra-estrutura são recebidos. Ele possui um conjunto de funções que permite receber o status das máquinas clientes, saber do estado das máquinas virtuais bem como operá-las, ou seja, iniciar e finalizar qualquer máquina virtual cadastrada no sistema.

5.6.1 Recebimento e tratamento das informações

Para monitorar o funcionamento das máquinas físicas o módulo servidor faz ciclicamente com intervalos de tempo pré-estabelecidos chamadas aos clientes requisitando o objeto máquina que contém um *snapshot* da máquina no momento. De posse desse objeto o servidor sabe todas as informações pertinentes sobre o cliente, ele faz um tratamento dessas entradas convertendo os valores obtidos para melhorar a visualização e atualiza as informações de cada máquina no banco de dados.

O monitoramento funciona da seguinte forma: um grupo de *threads* é iniciado e cada uma delas é responsável por fazer a requisição a uma máquina cliente, tratar os dados de retorno e atualizar as informações no banco de dados. Contudo essas *threads* não são lançadas todas juntas, um *pool* de *threads* é utilizado para iniciá-las em grupos e evitar um possível congestionamento na execução ou no acesso ao banco de dados.

A figura 5.11 demonstra como é obtido a informação das máquinas clientes pleo servidor e atualizadas no banco de dados. O servidor cria uma *thread* para cada máquina cliente, que se comunica com a máquina cliente, pega as informações, as trata e depois as envia para o banco de dados.

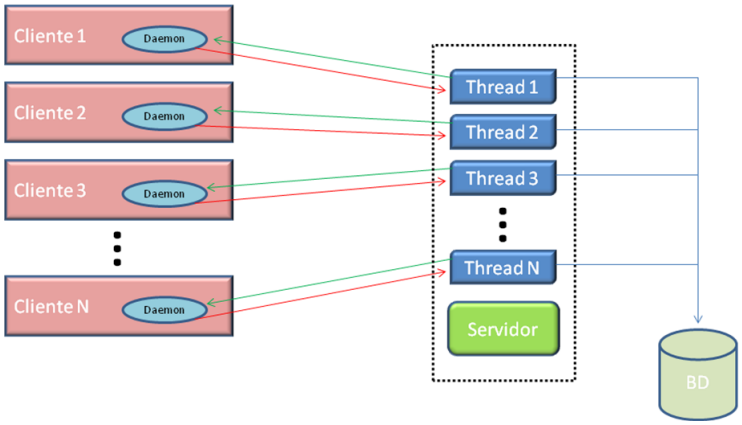

Figura 5.11: Comunicação entre o cliente e um servidor.

5.6.2 Operação das Máquinas Virtuais

A operação das máquinas virtuais que estão nas máquinas clientes funciona basicamente da mesma forma que a coleta das informações. O servidor faz uma requisição ao cliente para iniciar uma máquina virtual, por exemplo. Quando o cliente terminar a operação dessa atividade, ele deve retornar ao servidor se a operação foi bem sucedida ou não.

Quando a máquina virtual é iniciada com sucesso seu status é alterado para indicar que a mesma esta em operação, analogamente quando ela é encerrada o status dela deve ser alterado para sinalizar sua finalização. É através desses status que a camada do portal mostra aos usuários quais máquinas estão disponíveis para sua utilização, comentaremos mais sobre a interação entre o portal e o usuário na próxima sessão.

5.6.3 Reconhecimento de Falhas

Devido à natureza do ambiente que o *middleware* deve monitorar, as máquinas não estão totalmente sobre o poder dos operadores. A ocorrência de falhas nas máquinas, nas aplicações ou na comunicação são eventos frequentes e que devem ser tratados com muita atenção.

Para essa arquitetura, será considerada uma falha qualquer problema que impeça que a máquina esteja disponível para oferecer seus recursos. Para isso a aplicação cliente deve manter comunicação com o servidor constantemente, se o servidor fizer uma requisição de *status* e o cliente não responder, será considerada uma falha.

Quando ocorre a falha, a máquina é automaticamente desabilitada do sistema. Ela

para de ser monitorada e passa a não mais ser utilizada como parte da infra-estrutura até que ocorra uma intervenção humana e a recoloque novamente *on-line*.

Contudo, uma tolerância é aplicada para evitar falso negativo no reconhecimento de uma falha. A máquina só é efetivamente excluída da nuvem computacional se ela obtiver um número de falhas maior que um parâmetro pré-estabelecido de vezes. Essa abordagem foi escolhida para evitar a perda de poder computacional desnecessário. Uma máquina pode perder a conexão com o servidor por causa de um período de instabilidade da conexão, onde a comunicação seja restabelecida sem a necessidade de uma intervenção humana.

5.7 Comunicação Cliente-Servidor

A comunicação entre o cliente e o servidor é feita através da API RMI. Essa API é nativa do Java e sua finalidade é auxiliar no desenvolvimento de aplicações distribuídas. Ela é parte integrante do Java desde a versão JDK 1.1 e estendeu as funcionalidades da linguagem aos conceitos de programação distribuída. Seu princípio é permitir que um método pertencente a um objeto executando em uma JVM possa ser acessado por outra JVM, sendo que elas podem ou não estar rodando na mesma máquina física. A API provê uma camada de abstração que permite ao desenvolvedor se focar mais no desenvolvimento e alivia a preocupação quanto à comunicação e tratamento das conexões e etc.

A figura 5.12 mostra como esta comunicação está implementada internamente na JVM e como ela é realizada entre um cliente e um servidor.

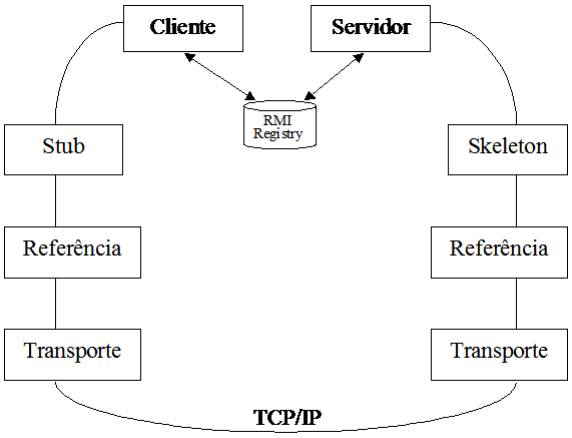

Figura 5.12: Exemplificação da comunicação entre o cliente e o servidor.

Em nosso trabalho, iremos utilizar a versão 1.6 do JDK. Nessa versão estão disponí-

veis importantes modificações que facilitaram o uso da API. Os *skeletons* não são mais necessários desde a versão 1.2 do JDK, um novo protocolo de *stub* foi criado que dispensa a necessidade de *skeletons* do lado servidor. A partir do JDK 1.5, os *stubs* são gerados automaticamente em tempo de execução.

Capítulo 6

Conclusão e Trabalhos Futuros

A necessidade de poder computacional é um ponto cada vez mais crítico na sociedade atual. A computação está presente para facilitar a vida da sociedade moderna em cada vez mais áreas. Podemos citar como alguns exemplos o uso da computação no poder público, áreas médicas, telecomunicação e diversas outras áreas. Esse aumento na demanda de poder computacional exige novas formas de computação e oferece um mercado promissor no oferecimento de computação como serviço.

Essa demanda e a popularização da *internet* são à base da computação em nuvem. Esse paradigma computacional promete fornecer alto poder computacional a usuários de forma simples e segura com um investimento muito menor do que seria gasto na compra e manutenção de uma infra-estrutura própria.

Nesse trabalho foram definidos os atributos essenciais para implantação de uma infra-estrutura de computação em nuvens e foram expostas suas características positivas bem como as principais dificuldades enfrentadas para sua implantação.

Foi também iniciado nesse trabalho um sistema gerenciador da nuvem computacional. Suas principais características seriam gerenciar de forma automática os recursos da nuvem computacional, coletando informações das máquinas físicas e detectando possíveis problemas que causem a indisponibilidade de recursos e operando as máquinas virtuais que serão oferecidas como recursos computacionais.

O portal tem o objetivo principal de ser a camada facilitadora para o uso dessa infra-estrutura complexa e heterogênea. É através dele que o usuário final gerencia os recursos que são de sua responsabilidade dentro da nuvem computacional e também utiliza todos os recursos que necessitar. Através de uma interface amigável, novos recursos podem ser inseridos e retirados por usuários com esses privilégios bem como podem ser utilizados

através de *desktops* dedicados somente de posse de um dispositivo de acesso a *internet*.

Com esse trabalho foi atingida diversas características do modelo IaaS. A virtualização dos recursos oferecidos para os usuários na forma de máquinas virtuais VMware. A Inclusão de novos recursos na nuvem computacional de forma *plug-and-play* é oferecida pelo portal com um cadastro simples da máquina hospedeira e das máquinas virtuais que ele pode oferecer. A requisição do recurso por um usuário é feita de forma bem direta e apesar da máquina virtual ser fixa em um hospedeiro, o usuário final não tem informações sobre a máquina física.

Algumas características de gerenciamento também são oferecidas. Através de um mecanismo de monitoramento o sistema pode retirar da infra-estrutura recursos que não estão respondendo por problemas técnicos ou de comunicação.

Além disso, um administrador da nuvem computacional ou de um *site* pode ligar ou desligar os recursos como quiserem pelo portal.

Contudo, outras características não foram atendidas por essa proposta. Os módulos de segurança tanto na comunicação entre o servidor e os recursos da nuvem computacional não estão sendo feitos de forma segura. Essa segurança será feita através da implementação de um *socket* seguro com comunicação SSL para a comunicação RMI. Já foram iniciados testes para realizar essa comunicação de forma segura, mas a solução ainda não foi incorporada.

A segurança no *login* dos usuários no portal seria feita através do uso de certificados devidamente assinados e reconhecidos nos navegadores que junto ao *login* autenticarão um usuário ao portal.

Apesar dos usuários já poderem utilizar o EasyGrid AMS manualmente entre as máquinas virtuais, a implantação nativa dessa ferramenta seria muito importante no oferecimento de poder computacional da nuvem computacional. Deve-se implantar uma interface simples que irá abstrair toda a geração dos arquivos de escalonamento necessários para execução do AMS.

Para as máquinas virtuais será implementado um servidor NIS e um NFS para centralizar os usuários que podem acessar as máquinas da nuvem computacional e seus respectivos espaços em disco de um *storage*. O NIS é um servidor de contas que será integrado com o portal permitindo que um cadastro único no portal permita seu acesso diretamente através de um terminal Ajax que também será implementado futuramente. O servidor NFS será onde os usuários colocarão seus aplicativos que rodarão na sua grade virtual, ele

é centralizado entre as máquinas virtuais e seus arquivos replicados através da montagem deste disco.

Referências

ANDERSON, D. et al. Seti@ home: an experiment in public-resource computing. *Communications of the ACM*, ACM, v. 45, n. 11, p. 61, 2002.

ARMBRUST, M. et al. *Above the Clouds: A Berkeley View of Cloud Computing*. United States: UC Berkeley Adaptative Distributed Systems Laboratory, fev. 2009.

BARHAM, P. et al. Xen and the art of virtualization. In: *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*. New York, NY, USA: ACM, 2003. p. 164–177. ISBN 1-58113-757-5.

BOERES, C. et al. An EasyGrid portal for scheduling system-aware applications on computational grids: Research articles. *Concurr. Comput. : Pract. Exper.*, John Wiley and Sons Ltd., Chichester, UK, v. 18, n. 6, p. 553–566, 2006. ISSN 1532-0626.

CARISSIMI, A. Virtualização: da teoria a soluções. In: *Anais do XXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. Rio de Janeiro, SP, Brasil: Sociedade Brasileira de Computação, 2008. p. 173–207.

EUCALYPTUS. 2010. Disponível em http://open.eucalyptus.com/. Última visita em 01 de Julho de 2010.

FOSTER, I.; KESSELMAN, C. Globus: A metacomputing infrastructure toolkit. *International Journal of Supercomputer Applications*, v. 11, p. 115–128, 1996.

FOSTER, I. et al. The Physiology of the Grid: An open grid services architecture for distributed systems integration. In: EDINBURGH. *Open Grid Service Infrastructure WG, Global Grid Forum*. [S.l.], 2002. v. 22, p. 1–5.

FOSTER, I. et al. Cloud computing and grid computing 360-degree compared. In: *Grid Computing Environments Workshop, 2008. GCE '08*. Austin, TX, USA: IEEE, 2008. p. 1–10. ISBN 978-1-4244-2860-1.

FOSTER, I. T. The anatomy of the grid: Enabling scalable virtual organizations. In: *Euro-Par '01: Proceedings of the 7th International Euro-Par Conference Manchester on Parallel Processing*. London, UK: Springer-Verlag, 2001. p. 1–4. ISBN 3-540-42495-4.

GEAMBASU, R.; GRIBBLE, S.; LEVY, H. Cloudviews: Communal data sharing in public clouds. *HotCloud. USENIX*, 2009.

GONÇALVES, D. B.; JUNIOR, J. C. V. *White Paper - Virtualização*. 2010. Disponível em http://www.sensedia.com/br/anexos/wp_virtualizacao.pdf. Última visita em 01 de Julho de 2010.

MENASCÉ, D. Virtualization: Concepts, applications, and performance modeling. In: CITESEER. *CMG-CONFERENCE-*. [S.l.], 2005. v. 1, p. 407.

MEZINI, F. V.-H. *Um Portal para Facilitar o Uso de Grades Computacionais*. Niterói: [s.n.], 2008.

MURPHY, A. *Virtualization Defined - Eight Different Ways*. 2010. Disponível em http://www.f5.com/pdf/white-papers/virtualization-defined-wp.pdf. Última visita em 01 de Julho de 2010.

NASCIMENTO, A. de P. *Conhecendo as Grades Computacionais*. Brasil: Revista Eletrônica de Computação - UniLaSalle, 2006.

NASCIMENTO, A. P. et al. Autonomic application management for large scale mpi programs. *Int. J. High Perform. Comput. Netw.*, Inderscience Publishers, Inderscience Publishers, Geneva, SWITZERLAND, v. 5, n. 4, p. 227–240, 2008. ISSN 1740-0562.

OPEN Cloud Manifesto - Dedicated to the belief that the cloud should be open. 2009. Disponível em http://www.opencloudmanifesto.org/opencloudmanifesto1.htm. Última visita em 01 de Julho de 2010.

OPENNEBULA. 2010. Disponível em http://www.opennebula.org/start. Última visita em 01 de Julho de 2010.

RODRIGUES, H. B. *Grid SA: Uma Sociedade Autônoma*. Dissertação (Mestrado) — Universidade Federal Fluminense, Niterói, 2009.

ROSENBLUM, M. The reincarnation of virtual machines. *Queue*, ACM, New York, NY, USA, v. 2, n. 5, p. 34–40, 2004. ISSN 1542-7730.

SANTOS, N.; GUMMADI, K.; RODRIGUES, R. Towards trusted cloud computing. *Proc. HotCloud*, 2009.

SILVA, J. A. *Tolerância a Falhas para Aplicações Autônomas em Grades Computacionais*. Tese (Doutorado).

SMEETS, B.; BONESS, U.; BANKRAS, R. *Programando Google Web Toolkit - Do Iniciante ao Profissional*. [S.l.]: Alta Books, 2009. ISBN 9788576083443.

STAINFORTH, D. et al. Distributed computing for public-interest climate modeling research. *Computing in Science and Engg.*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 4, n. 3, p. 82–89, 2002. ISSN 1521-9615.

VENITO, A. S. *MyEasyGridPortal: Uma abordagem para facilitar o acesso a múltiplas grades computacionais usando a tecnologia Portlet*. Dissertação (Mestrado) — Universidade Federal Fluminense, Niterói, 2006.

VMWARE Overview. 2010. Disponível em http://www.vmware.com/files/pdf/VMware-Company-Overview-DS-EN.pdf. Última visita em 01 de Julho de 2010.

WORLD Community Grid. 2010. Disponível em http://www.worldcommunitygrid.org/about_us/viewAboutUs.do. Última visita em 05 de Julho de 2010.

YEO, C. et al. Utility Computing and Global Grids. *Arxiv preprint cs/0605056*, 2006.

ZHANG, Q.; CHENG, L.; BOUTABA, R. Cloud computing: state-of-the-art and research challenges. *Journal of Internet Services and Applications*, Springer London, London, UK, v. 1, n. 1, p. 7–18, 2010. ISSN 1867-4828.