UNIVERSIDADE DO SUL DE SANTA CATARINA

JOHNATHAN SCHMITT

KLEBER LEONARDO DAMASCO

DESENVOLVIMENTO DE UM SERVIÇO UTILIZANDO AS MODALIDADES IAAS E PAAS PARA CLOUD COMPUTING: UM ESTUDO DE CASO

Palhoça

2012

JOHNATHAN SCHMITT
KLEBER LEONARDO DAMASCO

DESENVOLVIMENTO DE UM SERVIÇO UTILIZANDO AS MODALIDADES IAAS E PAAS PARA CLOUD COMPUTING: UM ESTUDO DE CASO

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação da Universidade do Sul de Santa Catarina, como requisito parcial à obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Flávio Ceci, M.Eng.

Palhoça
2012

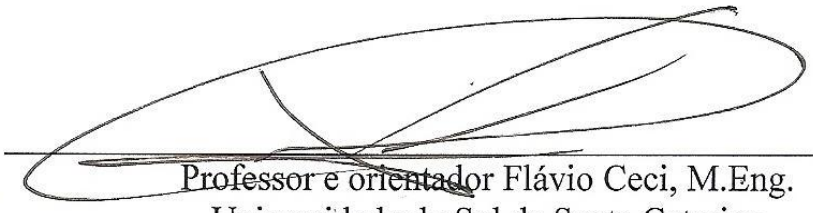
JOHNATHAN SCHMITT

KLEBER LEONARDO DAMASCO

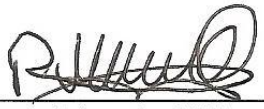
DESENVOLVIMENTO DE UM SERVIÇO UTILIZANDO AS MODALIDADES IAAS E PAAS PARA CLOUD COMPUTING: UM ESTUDO DE CASO

Este Trabalho de Conclusão de Curso foi julgado adequado à obtenção do título de Bacharel em Sistemas de Informação e aprovado em sua forma final pelo Curso de Graduação de Sistemas de Informação da Universidade do Sul de Santa Catarina.

Palhoça, 22 de novembro de 2012.



Professor e orientador Flávio Ceci, M.Eng.
Universidade do Sul de Santa Catarina



Professor Ricardo Villarroel Dávalos, Dr. Eng.
Universidade do Sul de Santa Catarina

Professor Roberto Fabiano Fernandes, M.Eng
Universidade do Sul de Santa Catarina

RESUMO

Com o grande crescimento da internet e dos acessos por dispositivos móveis vem à tona o conceito de computação nas nuvens, o qual busca fornecer capacidades computacionais e armazenamento de dados trazendo consigo a não dependência de um único recurso físico e, ao mesmo tempo garantindo a segurança, disponibilidade, elasticidade e performance. Deste modo veio o objetivo deste trabalho no qual visa desenvolver um protótipo de um tipo serviço, e assim adaptando-o as modalidades Infraestrutura como Serviço (IaaS - *Infrastructure as a Service*) e Plataforma como Serviço (PaaS - Platform as a Service) de modo a identificar as suas principais diferenças no processo de desenvolvimento e implantação. Para realizar a comparação entre as modalidades foi escolhido a estrutura da Google que utiliza a modalidade de PaaS e a da Amazon que utiliza a modalidade de IaaS, para validar os ambientes foi elaborado uma validação com as características essenciais que um serviço de computação nas nuvens deve prover, de tal forma que as duas atenderam a todas as características levantadas, mas tendo suas peculiaridades de acordo com cada modalidade. A partir do estudo de caso chegou-se à conclusão que o melhor ambiente para a aplicação do nosso protótipo seria o do Amazon, pois, além de fornecer o total controle sobre a máquina criada e as características deste ambiente não impõe limitações impactantes que a Google proveu.

Palavras-chave: Cloud Computing, IaaS, PaaS, SOA.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE SIGLAS

API – APPLICATION PROGRAMMING INTERFACE
BAAS – BACKEND AS A SERVICE
CAAS – COMPUTING AS A SERVICE
CRM – CUSTOMER RELATIONSHIP MANAGEMENT
DAAS – DATABASE AS A SERVICE
EAAS – EVERYTHING AS A SERVICE
EC2 – ELASTIC COMPUTE CLOUD
GAE – GOOGLE APP ENGINE
HTTP – HYPERTEXT TRANSFER PROTOCOL
IAAS – INFRASTRUCTURE AS A SERVICE
IBM – INTERNATIONAL BUSINESS MACHINES
IDE – INTEGRATED DEVELOPMENT ENVIRONMENT
JDO – JAVA DATA OBJECT
JPA – JAVA PERSISTENCE API
JSF – JAVA SERVER FACES
JSP – JAVA SERVER PAGES
MVC – MODEL VIEW CONTROLLER
NIST – NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
OMG – OBJECT MANAGEMENT GROUP
PAAS – PLATFORM AS A SERVICE
PC – PERSONAL COMPUTER
PDA – PERSONAL DIGITAL ASSISTANT
RUP – RATIONAL UNIFIED PROCESS
SAAS – SOFTWARE AS A SERVICE
SCP – SECURE COPY
S3 – SIMPLE STORAGE SERVICE
SAD – SISTEMAS DE ARQUIVOS DISTRIBUÍDOS
SDK – SOFTWARE DEVELOPMENT KIT

SGBD – SISTEMA DE GERENCIAMENTO DE DADOS

SLA – SERVICE LEVEL AGREEMENT

SOA – SERVICE ORIENTED ARCHITECTURE

SSH – SECURE SHELL

TAAS – TESTING AS A SERVICE

TCC – TRABALHO DE CONCLUSÃO DE CURSO

TI – TECNOLOGIA DA INFORMAÇÃO

UML – UNIFIED MODELING LANGUAGE

XML – EXTENSIBLE MARKUP LANGUAGE

XP – EXTREME PROGRAMMING

SUMÁRIO

1 INTRODUÇÃO

O surgimento da internet revolucionou o mundo da comunicação e dos dispositivos como, celulares, computadores, tablets. A internet por ter um alto poder de transmissão de dados, torna possível a interação entre as pessoas e os mais diversos dispositivos sem que tenha a limitação de distância entre eles. Ela é produto de grandes investimentos, pesquisas e desenvolvimento de infraestrutura para o compartilhamento de informações. (LEINER, et al, 2012, tradução nossa).

Segundo Cearley (2010, tradução nossa), vice-presidente do Gartner Group, *Cloud Computing* é um novo conceito de computação escalável, ou seja, os recursos necessários são fornecidos como um serviço, e os clientes precisam apenas da internet para poder utilizá-lo. Deste modo serviço pode ser considerado como transações que ocorrem entre o cliente e o prestador de serviço. "Os clientes não precisam saber como funciona, eles simplesmente poderão utilizar os serviços oferecidos".

Para Taurion (2009), "Este modelo vem com o propósito de implementar o conceito já existente de virtualização, de forma que inúmeros computadores possam ficar interligados constituindo uma poderosa máquina."

Neste novo modelo os consumidores finais têm como principal beneficio uma computação de forma escalável e mais econômica, pois o cliente só paga exatamente pelo tempo de uso e caso seja necessário mais recursos, os mesmos serão alocados de forma automática. Desta maneira, *Cloud Computing* é um novo e promissor paradigma computacional no qual a forma de armazenamento, processamento e aplicativos estarão em algum lugar na rede sob a responsabilidade total do provedor deste serviço e acessível de forma remota pela internet.

Este trabalho tem como foco estudar quais componentes são necessários para o desenvolvimento de aplicações utilizando conceitos de computação em nuvens e também identificar arquiteturas que facilitem este desenvolvimento. Nos próximos capítulos – através de um estudo de caso - serão apresentados todos os detalhes necessários para a utilização destes serviços.

1.1 PROBLEMÁTICA

Com a popularidade dos dispositivos móveis e o com facilidade de acesso aos computadores e notebooks, as informações estão cada vez mais ao alcance das pessoas em qualquer lugar e a qualquer hora. Cada vez mais as aplicações estão passando de *software standalone* para aplicações distribuídas ou *on-line*. Isso se deve ao fato de essas aplicações não terem uma portabilidade abrangente a todos os tipos de equipamentos, pois necessitam que a instalação seja feita em cada computador/dispositivo e amarradas às limitações dos sistemas operacionais e ao *hardware*. (TEIXEIRA, 2010).

Em uma aplicação local um dos problemas mais comum e bastante significativo é o de fazer a atualização do sistema, nesta situação é necessário estar removendo a versão antiga e colocando a versão mais nova em cada dispositivo onde esta aplicação está instalada. Isto demanda uma grande equipe, e um tempo absurdo em relação a uma aplicação nas nuvens, onde sua atualização seria feita em apenas uma máquina, de forma simples e rápida.

Outro problema recorrente a esse tipo de aplicação seria identificar a melhor maneira de estar realizando a rotina de *backup* de dados.

Em relação aos *backups*, Taurion (2009, p.135) afirma que:

> Backup é um problema para maioria das empresas principalmente para as pequenas e médias que, por terem um staff técnico reduzido, não conseguem manter operando eficazmente um sistema especifico. Os backups são intensivos em capital (a empresa tem que comprar software específico e a mídia para backup) e demanda um elaborado processo de gerenciamento. Em consequência, muitas empresas não têm processos adequados de proteção e recuperação de dados para seus PCs e laptops.

O *Cloud Computing* tem a convergência entre a virtualização onde as aplicações são emuladas em um ambiente virtual, e a disponibilização de software onde as aplicações podem ser acessadas atreves da internet (JUNIOR, et al, 2010).

Tendo em vista todos esses problemas aqui apresentados às organizações que optam pelo Cloud Computing como solução, se encontram com outro grande problema identificar em qual modalidade, Plataforma como Serviço (*Platform as a Service - PaaS*) ou Infraestrutura como Serviço (*Infrastructure as a Service - IaaS*), melhor vai se adequar as suas necessidades.

Por este último motivo formulou-se a seguinte pergunta de pesquisa:

O que é preciso ser alterado em um serviço para utilizá-lo a partir das modalidades IaaS e PaaS ?

1.2 OBJETIVOS

Os objetivos desta monografia são apresentados em objetivo geral e objetivos específicos.

1.2.1 Objetivos Gerais

Desenvolver um protótipo de serviço, adaptando-o as modalidades IaaS e PaaS de modo a identificar as suas principais diferenças no processo de desenvolvimento e implantação.

1.2.2 Objetivos Específicos

- Identificar e aplicar os ambientes disponíveis para *Cloud Computing*;
- Desenvolver um protótipo utilizando um ambiente web tradicional;
- Verificar as mudanças estruturais no desenvolvimento do protótipo em relação com a abordagem tradicional;
- Validar a implantação do protótipo segundo as características do *Cloud Computing*;

1.3 JUSTIFICATIVA

Existe a vantagem do baixo custo para se manter um ambiente em cloud computing que é a existência do conceito "*pay as you go*" (pagar pelo que se consome), desta maneira só é pago os recursos computacionais usados e, se houver a necessidade, pode-se aumentar os mesmos.

Quando há necessidade de se realizar armazenamento de dados em um ambiente nas nuvens só se paga pela quantidade de informações armazenadas, diminuindo assim os gastos gerados com backup.

Conforme Taurion (2009) os benefícios de um processo regular de *backup* nas nuvens, não requer investimentos em mídias e *software* específicos, e também tendo a vantagem de poder realizar a restauração dos dados remotamente.

Ao se manter um ambiente em *cloud* podem-se ter inúmeras vantagens em relação ao um ambiente *standalone*, ou seja, uma aplicação que deve ser instalada em cada computador que necessite acesso as suas funcionalidades, tendo em vista que um ambiente nas nuvens possibilita que o acesso às suas informações seja feito de qualquer lugar, além de que as atualizações do sistema podem ser realizadas em um único local, resultando um melhor aproveitamento de tempo e diminuição de mão de obra. De acordo com Camargo Junior et al. (2010, apud IBM, 2009 b):

> A computação em nuvens tem o potencial de reduzir o custo dos trabalhos de configuração, operação, gestão e monitoramento de sistemas em 50%. Pode ainda melhorar a utilização de capital da companhia em 75%, ao reduzir significativamente os custos de licença, diminuir os tempos de ciclo de provisionamento de gastos das empresas de semanas para minutos, melhorar a qualidade de seus aplicativos através da eliminação de 30% dos defeitos dos *softwares* e, por fim, reduzir os custos de suporte aos usuários finais (*help desk*) em até 40%.

Com os estudos das modalidades IaaS e PaaS, podem ajudar as organizações a escolher qual modelo melhor se adapta as necessidades de migração das soluções web para um ambiente nas nuvens.

1.4 ESTRUTURA DA MONOGRAFIA

O trabalho está dividido em seis capítulos e estes encontram-se descritos a seguir.

Capítulo 1: Apresenta o tema, objetivos, justificativa, problemática e a estrutura do trabalho;

Capítulo 2: Descreve o referencial teórico do trabalho, apresentando o desenvolvimento tradicional de software, a arquitetura orientada a serviço (SOA) e computação nas nuvens.

Capítulo 3: Define o tipo de pesquisa que é realizado nesta proposta, etapas metodológicas, cronograma, proposta da solução e as principais delimitações.

Capítulo 4: Apresenta a modelagem do protótipo proposto.

Capítulo 5: Descreve o processo de preparação dos ambientes IAAS e PAAS, bem como o processo de adaptação do protótipo e como se deu a sua implementação.

Capítulo 6: Define as conclusões e trabalhos futuros.

2 REFERENCIAL BIBLIOGRÁFICO

Este capítulo aborda a revisão bibliográfica para o desenvolvimento deste Trabalho de Conclusão de Curso (TCC). Neste capítulo são considerados os seguintes tópicos: desenvolvimento tradicional de *software,* Arquitetura Orientada a Serviços (SOA), computação nas nuvens e seus modelos para desenvolvimento como: *Software as a Service* (SaaS), *Infrastructure as a Service* (IaaS), *Platform as a Service* (PaaS).

2.1 DESENVOLVIMENTO TRADICIONAL DE SOFTWARE

A partir do momento que passa a utilizar computadores para resolver problemas, a capacidade do *hardware* dos equipamentos atualmente disponíveis no mercado vem crescendo rapidamente. Mais declaradamente, os computadores, notebooks e dispositivos móveis estão ficando cada vez mais rápidos e menores e, mesmo assim, o seu custo vem decrescendo. Como decorrência disso, a demanda por recursos cada vez mais elaborados nos sistemas de software tem também crescido de forma sólida, o que leva a um aumento na complexidade do desenvolvimento de *software,* com consequente aumento do custo.

Segundo Mendes (2002), aproximadamente há quatro décadas, o *software* compunha uma quase insignificante parcela quando confrontado com o *hardware*. Os custos para desenvolver e fazer a manutenção destes sistemas eram muito baixos, entretanto, atualmente os *softwares* têm dominado os sistemas computacionais.

Desenvolvimento de aplicações web apresentam mudanças expressivas em relação ao desenvolvimento de aplicações convencionais. O contexto varia desde diferenças técnicas até mesmo as organizacionais. Diferenças técnicas constituem as arquiteturas e tecnologias particulares empregadas e os impactos envolvidos. Já as organizacionais estão relacionadas a um uso estratégico dessas aplicações propondo melhorar o negócio.

Esta foi uma das primeiras mudanças a ser aplicada no desenvolvimento de novas aplicações. Anteriormente as aplicações eram em sua maioria desenvolvidas com o propósito

de serem hospedas em cada máquina onde houvesse sua necessidade de utilização, tornando de certa forma muito difícil e inviável as devidas atualizações, correções e manutenções destas aplicações já que deveriam ser feitas em cada dispositivo.

Para reduzir os impactos causados por essas mudanças, uma nova opção foi o desenvolvimento de aplicações web, que trás em sua estrutura uma conectividade de multiusuários sem que haja a necessidade deste aplicativo estar fisicamente instalado em seu dispositivo.

Esse modelo tradicional de desenvolvimento de software que é composto por sua implementação juntamente com seu contrato de manutenção vem perdendo forças com a entrada desse novo paradigma que é o modelo de computação em nuvens, isto porque a rápida ascensão da tecnologia vem nos propondo novos modelos de desenvolvimento, abrindo novas oportunidades e aumentando as possibilidades de desenvolvimento neste novo modelo computacional.

Com a entrada deste novo modelo, as aplicações ficaram hospedadas em algum lugar da internet, possibilitando acesso a multiusuários a qualquer lugar e em qualquer momento, pagando apenas pelo seu uso.

2.2 ARQUITETURA ORIENTADA A SERVIÇOS (SOA)

A Arquitetura Orientada a Serviço (SOA, Service Oriented Architecture), consiste em uma arquitetura de desenvolvimento que visa aprimorar a eficiência, a agilidade e a produtividade da empresa, fazendo com que os serviços sejam os principais meios de solução lógica para a realização dos objetivos estratégicos associados à computação orientada a serviço. (ERL, 2008).

A abordagem do SOA neste trabalho se deu ao fato que ele trabalha com o serviço como a modalidade SaaS da computação nas nuvens, onde o SOA é um modelo de produção que lida com a concepção e construção de software aplicando os princípios de computação orientada em serviços para a construção do software, enquanto que o SaaS é um modelo para a venda e distribuição de aplicações de software.

Como o conceito de SOA é complexo e abrangente, tem-se na Tabela 1 algumas considerações incorretas a respeito do SOA (FRONDANA et al, 2009).

Quadro 1 – Definições erradas de SOA.

Definições equivocadas de SOA	**Porque está incompleto/incorreto?**
SOA é uma maneira de alinhar Tecnologia da Informação (TI) e negócios	Não é apenas isso, apesar de ser um dos objetivos de SOA. Os sistemas fracamente acoplados que são obtidos com uma boa arquitetura SOA pode dar a agilidade necessária para tornar o alinhamento entre TI e negócios uma realidade. Em outras palavras, SOA objetiva a tomada de decisões de negócios suportadas pela tecnologia, e não restritas pela mesma.
SOA é qualquer programa com interface web	Errado. É possível implementar SOA com outras tecnologias, tal como a *Open Services Gateway Initiative*, que é uma plataforma de serviços baseada em Java e pode ser remotamente gerenciada, e existem aplicações web, tal como a Remote *Procedure Call*, que possibilita a chamada de procedimentos de uma máquina em outra, que fogem do escopo de SOA.
SOA é uma estratégia de reuso	Reuso faz SOA parecer tentador, entretanto quanto maior a granularidade de um componente mais difícil é reusá-lo. Apesar disso, SOA vai permitir a evolução dos serviços ao longo do tempo para adaptação a

	novas necessidades sem a necessidade de reprojetar programas.
SOA é uma solução *off the shelf*	SOA não é algo comprável, mas sim uma forma de arquitetar sistemas distribuídos que envolvam integração e distribuição.

Fonte: (Frondana et al , 2009).

De acordo com Frondana (et. al, 2009):

> Arquitetura Orientada a Serviços pode ser entendida como um estilo arquitetural para a construção de sistemas baseados em componentes modularizado, autônomos e fracamente acoplados, denominados serviços. Cada serviço expõe processos e comportamentos, através de contratos, que são compostos de mensagens em endereços detectáveis chamados terminais. O comportamento dos serviços obedece a uma política, externa ao próprio serviço.

Os serviços são "programas de software fisicamente independentes, com características de design distintas que dão suporte à obtenção dos objetivos estratégicos associados à computação orientada a serviços" (ERL, 2008).

Conforme Erl (2008), o serviço pode ser classificado como:

- Serviço entidade: representa um serviço centralizado no negócio, que fundamenta o contexto e o limite funcional em uma ou mais entidades de negócios relacionadas, sendo considerado um serviço altamente reutilizável.
- Serviço-tarefa: com limite diretamente associado a uma tarefa ou a um processo especifico de uma empresa controladora, tende a ter um menor potencial de reuso.
- Serviço utilitário: dedicado a fornecer funcionalidades reusáveis, como registro de logs, notificações e tratamento de exceções, sendo conhecido com um serviço de aplicativo, de infraestrutura ou de tecnologia.

O SOA ajuda em muito as empresar de desenvolvimento a interagis os sistemas sem a necessidade de um software e interface especifica, tendo como proposta de conectar os sistemas por meio de interfaces abertas baseadas no padrão XML - Extensible Markup Language. (TAURION, 2007).

2.3 COMPUTAÇÃO NAS NUVENS

Esta seção está divida em subitens com o intuito de apresentar e detalhar alguns aspectos relevantes sobre computação nas nuvens. Tópicos como visão geral, conceitos, características e modelos de serviços são apresentados e detalhados a seguir.

2.3.1 Visão geral

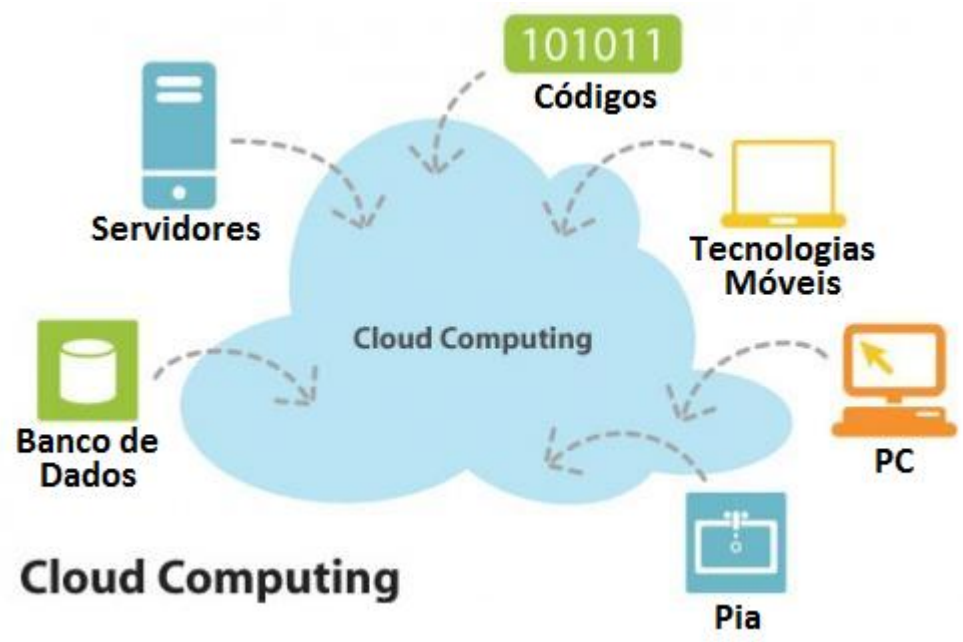
Com o progresso da sociedade moderna, os serviços básicos e essenciais tais como: água, gás, energia elétrica, no qual, são de utilidade pública são entregues a nós quase todos de uma maneira transparente. Estes serviços possuem infraestruturas compatíveis para tais serviços, tornando-os disponíveis em qualquer lugar a qualquer hora. Desta forma, o uso destes serviços pode ser cobrado apenas pelo seu uso de acordo com o tipo de tarifação viável ao usuário final. Esta mesma ideia, recentemente, vem sendo adotada e aplicada no contexto da informática, fazendo com que o termo comunicação nas nuvens venha a ser cada vez mais comum (VECCHIOLA et al. 2009, tradução nossa).

Computação em nuvem é uma forte tendência para o futuro não muito distante da Tecnologia da Informação (TI), pois esse novo modelo que vem emergindo fortemente e com o objetivo de fazer com que os serviços de TI sejam pagos por demanda, ou seja, baseados no uso (VECCHIOLA et al. 2009, tradução nossa).

Segundo Hayes (2009, Tradução nossa), comunicação nas nuvens tem se tornado muito popular e está associada à utilização da rede mundial de computadores com o uso massivo de servidores físicos ou virtuais, a nuvem, por exemplo, para a alocação de um ambiente de computação. Esse novo modelo computacional permite o acesso, sob demanda e através da *internet*, a um *pool* de recursos computacionais (redes, servidores, armazenamento, aplicações e serviços) que podem ser rapidamente provisionados.

A Figura 1 ilustra a visão geral da comunicação nas nuvens.

Figura 1– Nuvem Computacional.



Fonte: (TECHMIXER, 2010, Adaptado Pelos Autores).

Com isso, comunicação nas nuvens torna-se palavra chave da indústria de TI, pois este novo modelo computacional possibilita que seus usuários movam seus dados e aplicações para a nuvem e desta forma deixando-o acessível de qualquer dispositivo que possua um sistema operacional com internet e um *browser,* isso tudo de forma transparente para quem usa, ou seja, o usuário não precisa ter o conhecimento da tecnologia utilizada.

De maneira geral o entendimento sobre esse novo modelo computacional ainda não está bem claro, pois percepções errôneas estão se popularizando, segundo Taurion (2009) algumas delas são:

- A computação nas nuvens não será uma alternativa à internet, no entanto, terá sua abrangência aumentada com o conjunto de serviços que esse modelo proporciona;
- Esse modelo computacional não é uma fonte de recursos gratuitos, pois seu uso está condicionado à aderência as políticas e critérios de acesso, contabilização e gerenciamento;
- Os *data centers* atuais não estão com os dias contados, provavelmente as pequenas e médias empresas deixem de operar seus atuais data centers, no entanto as grandes empresas além de continuarem a usar seus data centers

também vão poder adotar esse modelo e oferecer serviços computacionais ao mercado;

- Não é possível usar livremente os recursos de diferentes nuvens, pois não existe essa interoperabilidade entre a maior parte das nuvens. Caso seja necessário integrar uma aplicação de e-mail de uma nuvem com uma aplicação em outra nuvem será muito complicado ou até mesmo impossível neste momento atual, não sendo descartável essa funcionalidade com o amadurecimento desse mercado;
- Não é tão simples trocar de nuvem já que vai exigir algum tempo para que recuperar e carregar os dados de uma nuvem para a outra. Será simples caso seja pessoa física devido a massa de dados não ser de certa forma elevada;
- Não será possível a troca do data center atual hoje para o modelo de nuvens amanhã tendo em vista que muitas nuvens restringem o tipo de linguagem e ambiente operacional a ser usado e nem sempre são os mesmos que você está utilizando em sua infraestrutura e aplicações.

O mesmo autor ainda afirma que esse modelo computacional irá provocar grandes impactos na maneira que as empresas usam TI e como esses fornecedores vendem TI e o compara com o impacto do *e-business* anos atrás, quando esse impacto mudou a visão e o papel da TI nas organizações.

2.3.2 Conceitos

Nesta seção são abordados os conceitos de comunicação nas nuvens segundo alguns autores.

A nuvem é um tipo sistema que possui vários computadores interconectados e virtualizados possíveis de serem dinamicamente provisionados e oferecidos com um ou

inúmeros recursos computacionais de acordo com o contrato do prestador do serviço e os consumidores (BUYYA, et al, 2008).

Para Taurion (2009), "Computação na nuvem implementa o conceito de virtualização, permitindo que inúmeros computadores interligados gerem a imagem de um poderoso supercomputador virtual." O mesmo autor ainda afirma que "[...] o conceito de Computação em Nuvem é um passo evolutivo na eterna busca pelo compartilhamento e consequentemente maior aproveitamento dos recursos computacionais." Esse modelo de computação é uma excelente alternativa para se criar um data center virtual, pois pode interligar milhares de servidores, internos e/ou externos através da rede reduzindo em larga escala o seu custo já que a computação em nuvem tem essa capacidade de maximizar e flexibilizar os recursos computacionais.

Segundo Bolsoni et al (2009), "[...] computação nas nuvens consiste no compartilhamento de dispositivos e ferramentas computacionais através da interligação dos sistemas, sempre disponíveis, em que não mais há ferramentas ou softwares locais [...]".

O *National Institute of Standards and Technology* (NIST), conceitua computação nas nuvens como um modelo que possibilita o acesso, de modo conveniente e sob demanda, a um conjunto de recursos computacionais configuráveis, por exemplo, redes, servidores, armazenamento, aplicações e serviços que são adquiridos e liberados de maneira rápida e fácil com um mínimo esforço gerencial ou interação com o provedor destes serviços (BADGER, et al, 2011, tradução nossa).

2.3.3 Características Essenciais

As características essenciais são as vantagens que esse modelo computacional oferece. Algumas destas características definem exclusivamente a computação em nuvem e faz a distinção com outros paradigmas. Segundo Leavitt (2009), Mell e Grance (2009) (tradução nossa) algumas características deste modelo são:

- Atendimento *self-service* e sob demanda. Um consumidor de um serviço de nuvem pode requisitar automaticamente (normalmente usando-se de interfaces de programação ou APIs) um recurso computacional (por

exemplo, um servidor virtual ou espaço em um servidor de armazenamento em rede).

- Elasticidade. O consumidor do serviço pode requisitar dinamicamente mais ou menos recursos para sua aplicação para se adaptar à demanda dos seus usuários. Por exemplo, em períodos de pico o consumidor solicita à nuvem mais recursos computacionais (como, por exemplo, servidores adicionais), podendo depois liberar tais recursos, quando os mesmos não forem mais necessários.
- Pagamento pelo uso e garantias de serviço (*Service Level Agreements – SLAs*). Os consumidores pagam aos provedores de serviço de nuvem de acordo com o consumo efetuado (modelo este semelhante ao utilizado com os serviços básicos como energia, água e gás). A nuvem possui mecanismos para permitir o compartilhamento dos recursos existentes de forma eficiente para os diversos clientes e monitorar as aplicações e os recursos de forma a respeitar as garantias de serviço oferecidas como, por exemplo, disponibilidade de 99,9%.
- Acesso ubíquo através da rede. Os recursos computacionais podem ser acessados através de padrões (como APIs REST baseadas em HTTP ou SOAP) por diversos tipos de clientes (*browsers*, PDAs, celulares) e seus detalhes de funcionamentos e complexidades ficam abstraídos pela nuvem.

Para Taurion (2009, p. 2) e Armbrust et al (2009, tradução nossa) essas são características bem relevantes da computação nas nuvens*:*

- A Computação em Nuvem cria uma ilusão da disponibilidade de recursos infinitos, acessíveis sob demanda;
- A Computação em Nuvem elimina a necessidade de adquirir e provisionar recursos antecipadamente;
- A Computação em Nuvem oferece elasticidade, permitindo que as empresas usem os recursos na quantidade que forem necessários, aumentando e diminuindo a capacidade computacional de forma dinâmica;
- O pagamento dos serviços em nuvem é pela quantidade de recursos utilizados (*pay-per-use*).

Segundo Gartner (2009, tradução nossa), cinco são os atributos que caracterizam a computação em nuvem:

- Ser baseado em serviço;
- Ser escalável e elástico;
- Ser compartilhado;
- Ser medido por uso;
- Usar tecnologias da Internet.

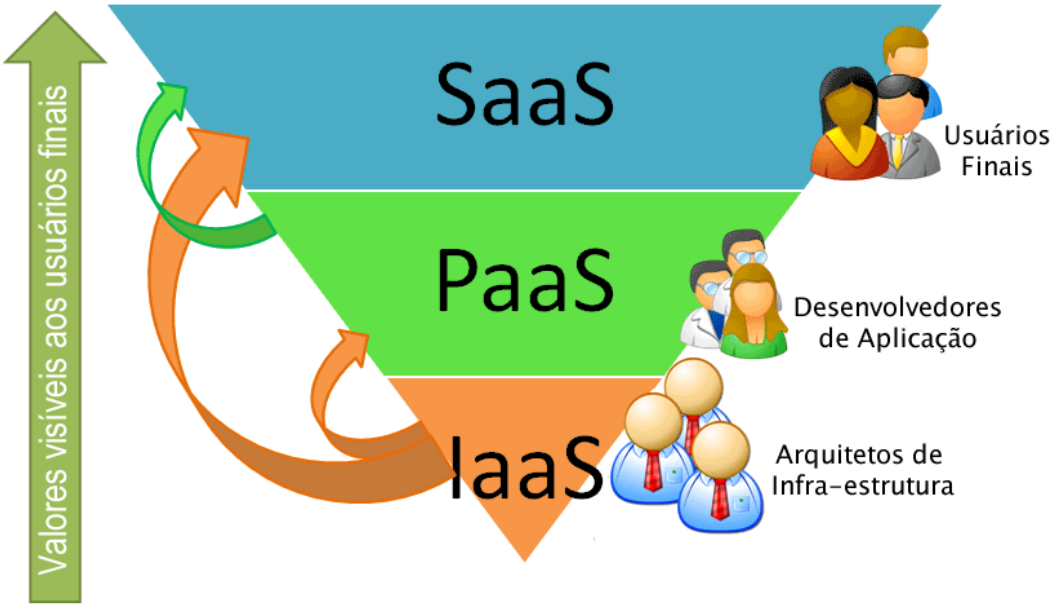
2.3.4 Modelos de Serviço

O ambiente de computação em nuvens está atualmente dividido em diversos modelos de serviços, e esse conceito de cada um desse serviço varia de acordo com o que cada companhia oferece. Alguns exemplos de serviços oferecidos atualmente no mercado são:

- *Backup as a Service (*BaaS*)* - Backup como um serviço*;*
- *Communication as a Service (*CaaS*)* - Comunicação como Serviço*;*
- *Database as a Service (*DaaS*)* - Banco de dados como um serviço*;*
- *Infrastructure as a Service (*IaaS*)* - Infraestrutura como Serviço*;*
- *Platform as a Service (*PaaS*)* - Plataforma como Serviço;
- *Software as a Service (*SaaS*)* - Software como Serviço*;*
- *Testing as a Service (*TaaS*)* - Teste como Serviço .
- *Everything as a Service (*EaaS*)* - Tudo como Serviço*;*

No entanto, para este estudo de caso são abordados e detalhados as seguintes modalidades de serviços: *SaaS*, *PaaS* e *IaaS.* A Figura 2 apresenta uma visão geral desses três tipos de ambientes, onde os arquitetos de infraestrutura montam a sustentação desse ambiente deixando um conjunto de ferramentas para o desenvolvimento destas aplicações tornando-as disponíveis ao usuários/clientes finais na ultima camada chamada de SaaS.

Figura 2 – Pirâmide representativa dos modelos de serviço.



Fonte: (SCHULLER, 2008, Adaptado Pelos Autores).

2.3.4.1 Software como Serviço (SaaS)

O modelo de SaaS proporciona *softwares* com finalidades específicas que estão disponíveis para múltiplos usuários/clientes finais sejam pessoas físicas ou empresas que possuem acesso a uma interface *thin client* como um navegador web, fazendo desta forma a diferenciação junto ao modelo tradicional que necessita que o usuário/cliente adquira a licença e precise instalar este aplicativo em quantos equipamentos forem necessários.

Segundo Taurion (2009, p.101), "O modelo SaaS muda as regras do jogo, transformando a maneira como o software é comercializado", tendo em vista que os usuários/clientes não têm mais a necessidade de ter o aplicativo na sua máquina, pois esse aplicativo estará na nuvem pronto para ser utilizado a qualquer momento e gerando o custo apenas quando o mesmo estiver utilizando. Ideia está confirmada por Taurion (2009, p.101), "[...] quando o software não está sendo usado, não está sendo pago".

Para MA (2007 apud Cancian, 2009) “O cliente possui direitos sobre seus dados e o de uso de software, mas em nenhum momento precisa adquirir uma licença ou comprar o software como se fosse um produto”.

Outro aspecto a ser observado no SaaS é que esses usuários/clientes finais não precisam ter o conhecimento do tipo de tecnologia/arquitetura que fora utilizado para o desenvolvimento deste aplicativo, pois os mesmos não irão fazer o controle da infraestrutura que está acerca deste aplicativo, por exemplo, seus servidores, sistemas operacionais e etc.

A Figura 3 ilustra a macro-visão do SaaS, de como os usuários/clientes se conectam a esses serviços.

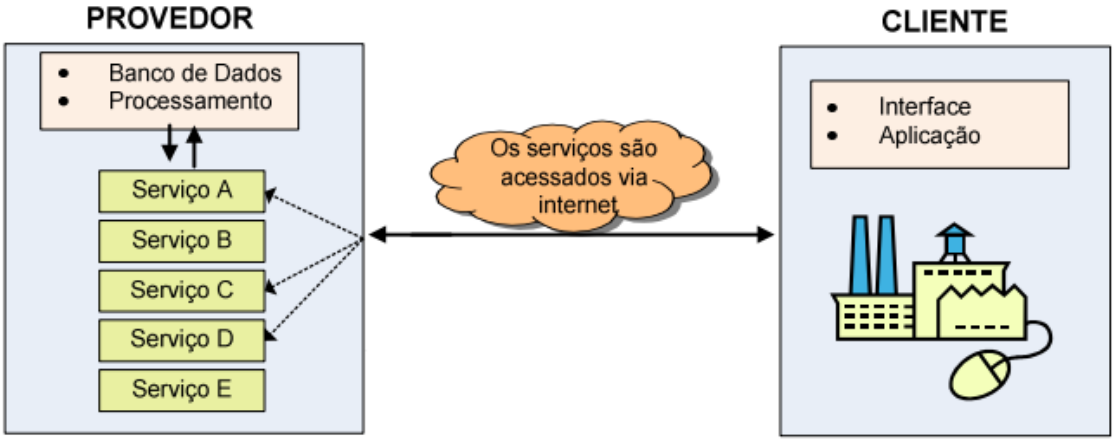
Figura 3 – Modelo Saas.

Fonte: (FERMO, 2012, Adaptado Pelos Autores).

A Figura 3 ilustra o funcionamento do SaaS, o aplicativo está na nuvem, acessível através de qualquer dispositivo que possua internet e um *browser*, isso tudo de forma clara e limpa para o usuário/cliente que não precisa saber de qual maneira é seu funcionamento.

A Figura 4 ilustra a visão do serviço SaaS e seu real funcionamento.

Figura 4 – Visão Saas.



Fonte: (CANCIAN, 2009, Adaptado Pelos Autores).

A Figura 4 ilustra o funcionando desta nuvem, que na situação fica incumbida de realizar todos os serviços necessários para atender todas as solicitações advindas dos seus usuários/clientes.

Conforme Eliadis e Rand (2007 apud Cancian 2009), "O mercado de software SaaS vem amadurecendo e crescendo rapidamente e tem atraído fornecedores de diferentes segmentos do mercado global de TI, bem como um conjunto diverso de clientes."

Segundo previsões de especialistas mencionadas por Clair (2008), eles afirmam que 35% do software aplicativo a ser implementado até 2012 serão fornecidos via SaaS, que 25% do orçamento de Tecnologia da Informação serão destinados a aplicações SaaS em 2013.

Uma empresa bastante conhecida por fornecer esse tipo de serviço é a *SalesForce*, que oferece um dos *softwares* mais reconhecidos do mercado, o *Customer Relationship Management* (CRM). Outro marcante exemplo que atualmente é a *Google*, que vem investindo nessa tecnologia e que disponibiliza aos seus usuários o *Google Docs*, um pacote Office que não necessita instalação nem a compra destes aplicativos.

O *Microsoft Azure* também é um exemplo de SaaS, que tem o objetivo de prover serviços que execute nas nuvens, ele contem ferramentas para que possa ser feito a criação, implantação e gerenciamento destes aplicativos (WINDOWS AZURE, 2012).

Segundo Armbrust et al (2009, tradução nossa) as vantagens que esse modelo apresenta são muito bem compreendidas para os usuários finais e para os prestadores deste serviço, pelo simples fato que a manutenção e atualizações ficam centralizadas e os usuários/clientes podem acessar a qualquer hora e em qualquer lugar.

Os benefícios esperados e que atualmente estão sendo demonstrados com o uso do *SaaS* na prática, segundo (Taurion 2009, p. 106) são:

- **Simplificação do gerenciamento dos aplicativos** (Instalação, upgrades e manutenção).
- **Redução do investimento em capital**. Empresas que estão enfrentando escassez de credito vão tentar manter seu caixa o mais intacto possível. O modelo SaaS reduz o investimento em capital e torna-se extremamente atraente nesse contexto.
- **Velocidade de implementação.** O processo de implementação é acelerado pois uma serie de atividades que não agregam valor, como instalação e configuração, simplesmente deixam de existir. Alem disso, os produtos *SaaS* usam interfaces baseadas em *browsers*, mais intuitivas e fáceis de usar.
- **Possibilitar que a área de TI se concentre em apoiar o negócio-fim da empresa.** E não desperdice tempo fazendo correções e upgrades de *software.*
- **Acesso a novas funcionalidades de forma rápida e sem o oneroso processo atual, de instalar uma nova versão.** Esse é um ponto importante. Muitas empresas acabam não atualizando suas versões de software e não obtendo as vantagens das novas funcionalidades, devido ao oneroso processo de atualização imposto pelo modelo tradicional.

No entanto, apesar deste modelo trazer inúmeras vantagens conforme supramencionadas, Taurion (2009, p. 107) alerta para alguns cuidados que devem ser observados pelas empresas que estão adotando o modelo SaaS*:*

1. Deve ser feita uma analise econômica e financeira bem cautelosa com o intuito de analisar os gastos totais pelo período em que o software estará em utilização pela empresa;
2. Analisar a empresa provedora do serviço para tentar prever possíveis problemas econômicos que afetariam a distribuição deste serviço;
3. Verificar os níveis de serviço que estarão disponíveis;
4. Geração de multas caso o sistema esteja indisponível.

O autor ainda afirma que mesmo sendo mais fácil a aquisição desses aplicativos SaaS, o processo para a obter a licença de utilização destes aplicativos, não pode deixar de passar por um processo formal dentro da área de TI.

Algumas desvantagens existem, e são elas que ajudam a dificultar a adoção a este novo modelo, Taurion (2009), fala sobre a localização destes dados armazenados na nuvem, no qual pode haver duvida legal por esses dados de cidadãos de um país estar residindo em outro, e afirma também que a segurança dos dados é fundamental, já que é necessário ter garantias sobre esses dados contidos nas nuvens de tal forma a garantir que somente serão acessados por usuários autorizados. Seguindo nessa mesma linha da segurança dos dados, Kaufman (2009, tradução nossa) completa que o provedor deste serviço deve oferecer no mínimo recursos que possam criptografar esses dados e que ainda possua datas previstas para fazer *backup* dos mesmos.

2.3.4.2 Plataforma como Serviço (PaaS)

Plataforma como Serviço é a camada intermediária da arquitetura de Computação nas Nuvens. Esta arquitetura define o ambiente onde será desenvolvida a aplicação (LAWTON, 2008, tradução nossa). Para Taurion (2009, p. 132), o modelo PaaS "se propõe a criar uma plataforma para o desenvolvimento de aplicações já voltadas para a Computação em Nuvem".

Esse modelo fornece um conjunto de ferramentas para que o cliente possa desenvolver, implantar e administrar o software aplicativo que está estruturado para suportar um grande número de assinantes, quantidade de processos muito grandes de dados, e, potencialmente ser acessado de qualquer ponto a *internet*. Nuvens PaaS normalmente fornecem um conjunto de ferramentas de desenvolvimento, tais como linguagens de programação, banco de dados e suporte em tempo de execução facilitando a construção de alta qualidade de aplicações escaláveis. Além disso, nuvens PaaS fornecem ferramentas que auxiliam a implantação de novas aplicações, pois esse modelo já vem preparado para você implantar sua aplicação sem grandes preocupações.

Uma plataforma que utiliza o modelo PaaS é o Google *App Engine*, que disponibiliza gratuitamente, com algumas limitações, um ambiente com *API's on demand* (SCHOFIELD, 2008, tradução nossa).

Segundo Lawton (2008, tradução nossa), a vantagem de desenvolver aplicativos nessa plataforma é que o programador não vai precisar se encarregar em determinar o comportamento dos diferentes sistemas operacionais, lidar com linguagens de programação e acesso a recursos específicos, pois a própria plataforma já fica encarregada desta tarefa.

No entanto algumas restrições são colocadas nesse modelo, segundo Taurion (2009), essas nuvens restringem por tipos de linguagens as aplicações que podem ser colocadas nelas, dois exemplos citados por ele seria o caso do PaaS do *Google* o *AppEngine* que atualmente oferece recursos apenas para o *Python*¸ e o caso da plataforma do *Salesforce.com* que permite apenas o uso de linguagem própria, criada por eles que é a *Apex*.

2.3.4.3 Infraestrutura como Serviço (IaaS)

A infraestrutura como um serviço é a camada base para todos os outros serviços desse modelo computacional sendo ela responsável por disponibilizar todos os recursos de hardware, ou seja, seus processadores, armazenamentos e servidores.

Este termo originalmente surgiu em 2006 como uma sucessão do conceito de *Hardware-as-a-Service*, recomendado pelo jornalista Nicholas Carr. Para Taurion (2009, p.100) "A ideia básica é que o usuário, em vez de adquirir e instalar servidores e equipamentos de rede em um data center, poderia usar estes recursos a partir de um provedor externo". Como exemplos deste serviço atualmente oferecidos são o EC2 e S3 da *Amazon*, o *Sun Grid* da *Sun Microsystems* o *Blue Cloud* da *IBM* e o *RackSpace*.

Este serviço é oferecido através da tecnologia de virtualização, no qual o provedor oferta diversos tipos de configurações e o cliente que escolhe o que melhor vai se adequar as suas necessidades, sendo que esses recursos oferecidos serão alocados conforme sua demanda. O cliente deste modelo não faz o controle da infraestrutura da nuvem, mas tem o

controle total dos recursos de sua máquina virtual onde o mesmo fica totalmente responsável por instalar tudo o que for necessário para sua aplicação.

Segundo Cancian (2009), o cliente pode adquirir estes recursos como serviço, totalmente terceirizado não precisando adquirir servidores de alto desempenho, aplicativos complexos e equipamentos de rede, no qual o seu pagamento deve ser contabilizado pela utilização desses recursos computacionais.

Segundo Taurion (2009, p. 100) as características básicas do modelo Infraestrutura como Serviço são:

a) O usuário não precisa dispor de hardware e software nos moldes tradicionais, ou seja, em seu data center. A capacidade de processamento e de armazenamento é obtida remotamente da nuvem.
b) Todos os recursos computacionais estão na nuvem do provedor, que os alocará de forma dinâmica e elástica, para atender às demandas de flutuação do negócio.
c) O acesso à nuvem e via internet. Portanto, banda larga e fundamental.
d) Todo o pagamento é pelo volume de utilização. Usou e pagou pelo que foi usado.

Esse modelo incentiva a criação de ecossistemas que podem gerar aplicações e serviços complementares à oferta de IaaS. Um exemplo é ecossistema criado em cima da nuvem da *Amazon*, com inúmeras *start-ups* disponibilizando serviços adicionais usando esta nuvem como infraestrutura.

2.3.5 Modelos de implantação

Corresponde o acesso aos ambientes disponibilizados na computação em nuvens, com os tipos diferentes de implementação e dependendo das regras de negócio. Assim dependendo da empresa e do contexto da aplicação as informações podem ser mais utilizadas em ambientes mais restritos (TAURION, 2009).

De acordo com Verdi (2010) e Badger (2011) há quatro maneiras para se implantar e disponibilizar os serviços de comunicação nas nuvens e são elas: nuvens privadas, nuvens públicas, nuvens comunitárias e nuvens híbridas. Mas segundo Naito (2010) a um quinto tipo de nuvem chamada de *intercloud*.

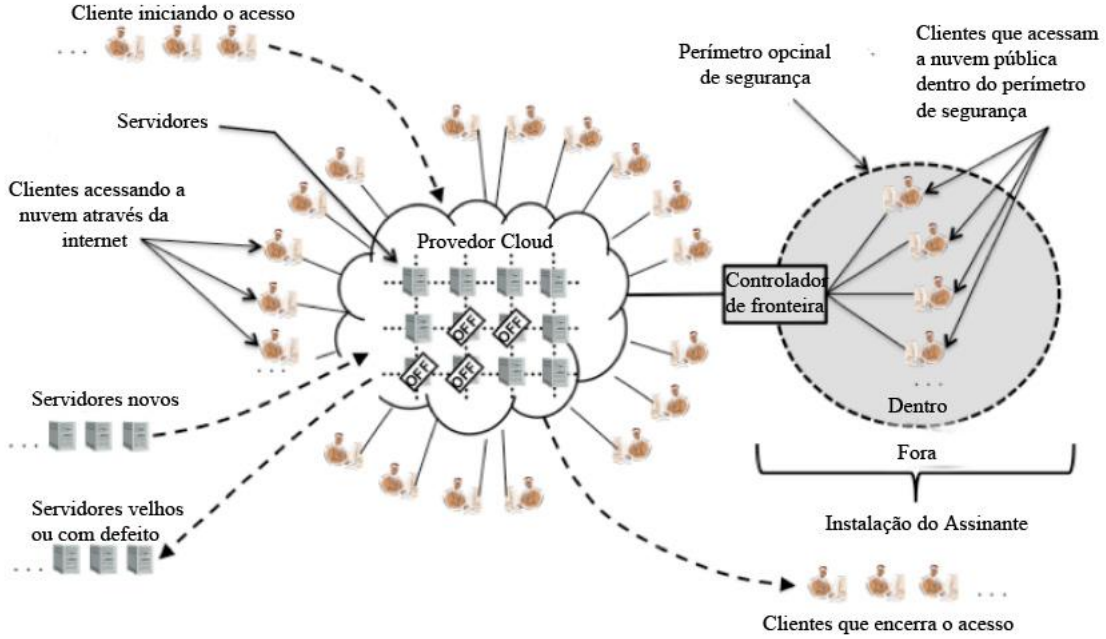
2.3.5.1 Nuvens públicas (*Public Cloud*)

A nuvem mais popular, mesmo sendo chamada de pública não significa necessariamente que a mesma é gratuita, dependendo do modelo de negócio adotado pelos provedores, sendo oferecido um rápido acesso a infraestrutura computacional com um custo mais baixo. Mas tendo com desafios a segurança, com a confiabilidade e portabilidade (TAURION, 2009).

Tendo como exemplo algumas aplicações que disponibilizam o modelo de nuvem pública (BASANT, 2011):

- *Google App Engine;*
- *Microsoft Windows Azure;*
- *IBM Smart Nuvem;*
- *Amazon EC2.*

Figura 5 – Nuvem Pública.



Fonte: (BADGER, 2011, Adaptado pelos autores).

A Figura 5 mostra o cenário de uma nuvem pública. De modo a se ter duas formas de se acessar esse modelo, um sendo pelo acesso a internet e o outro acesso sendo protegido por um perímetro de segurança.

2.3.5.2 Nuvens privadas (*Private Cloud*)

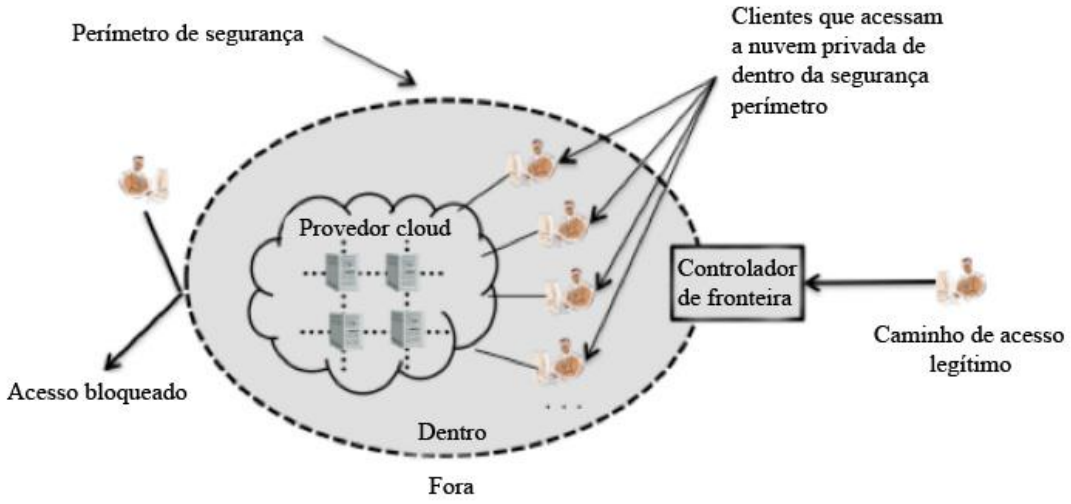
De acordo com Taurion (2009) a nuvem privada também é chamada de nuvem empresarial, a onde usa os conceitos de nuvem computacional aplicando nos servidores onde se encontram localizados entro do *firewall* das empresas.

"Nuvens privadas diferem das nuvens públicas em que a rede, computação e armazenamento de infraestrutura associada com nuvens privadas é dedicado a uma única organização e não é compartilhado com outras organizações." (POSSOBOM, 2010).

Tendo como exemplo algumas aplicações que disponibilizam o modelo de nuvem privada (BASANT, 2011):

- *Eucalyptus;*
- *Ubuntu Enterprise Cloud – UEC;*
- *Amazon VPC;*
- *VMware Cloud Infrastructure Suite;*
- *Microsoft ECI data center.*

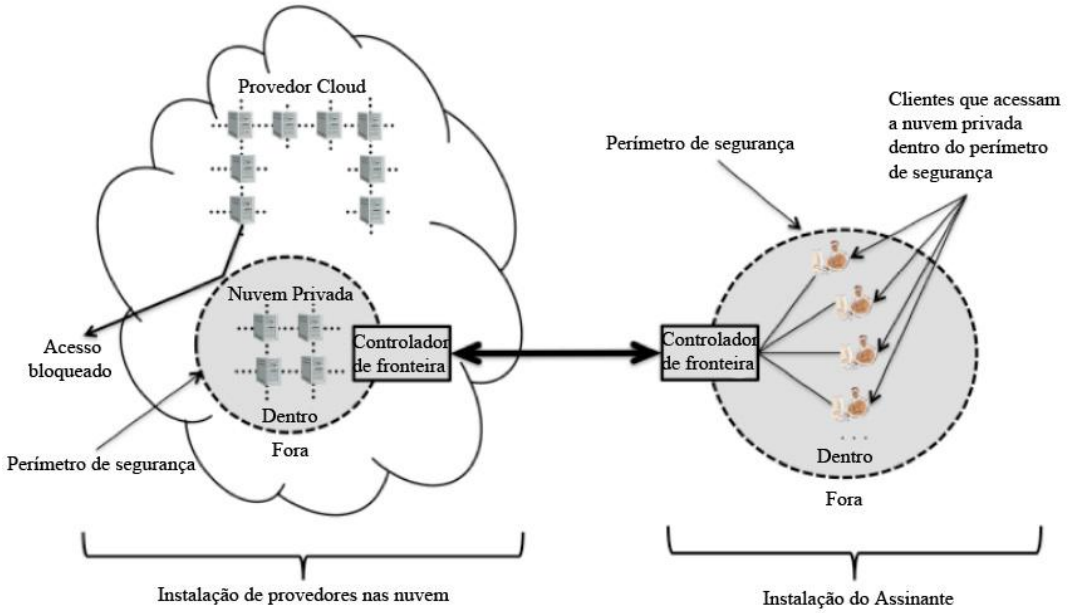
Figura 6 – Nuvem Privada On-Site.



Fonte: (BADGER, 2011, Adaptado pelos autores).

A Figura 6 ilustra um exemplo de uma nuvem pública on-site que consiste em uma nuvem implantada dentro das instalações de um cliente. Permite segurança em torno de todo dos assinantes e entorno da nuvem. De moda que para acessar essa nuvem de fora do perímetro de segurança tem que passar pelo controlador de fronteira, verificando se o usuário tem permissão para acessa a nuvem. (NAITO, 2010).

Figura 7 – Nuvem Privada Terceirizada.



Fonte: (BADGER, 2011, Adaptado pelos autores).

A Figura 7 ilustra o cenário de uma nuvem privada terceirizada onde os servidores são terceirizados por uma empresa hospedagem, de modo que a nuvem tem dois perímetros de segurança, no lado esquerdo implementada por um provedor e no lado direito implementada pelo assinante da nuvem (NAITO, 2010).

2.3.5.3 Nuvens Comunitárias (*Community Cloud*)

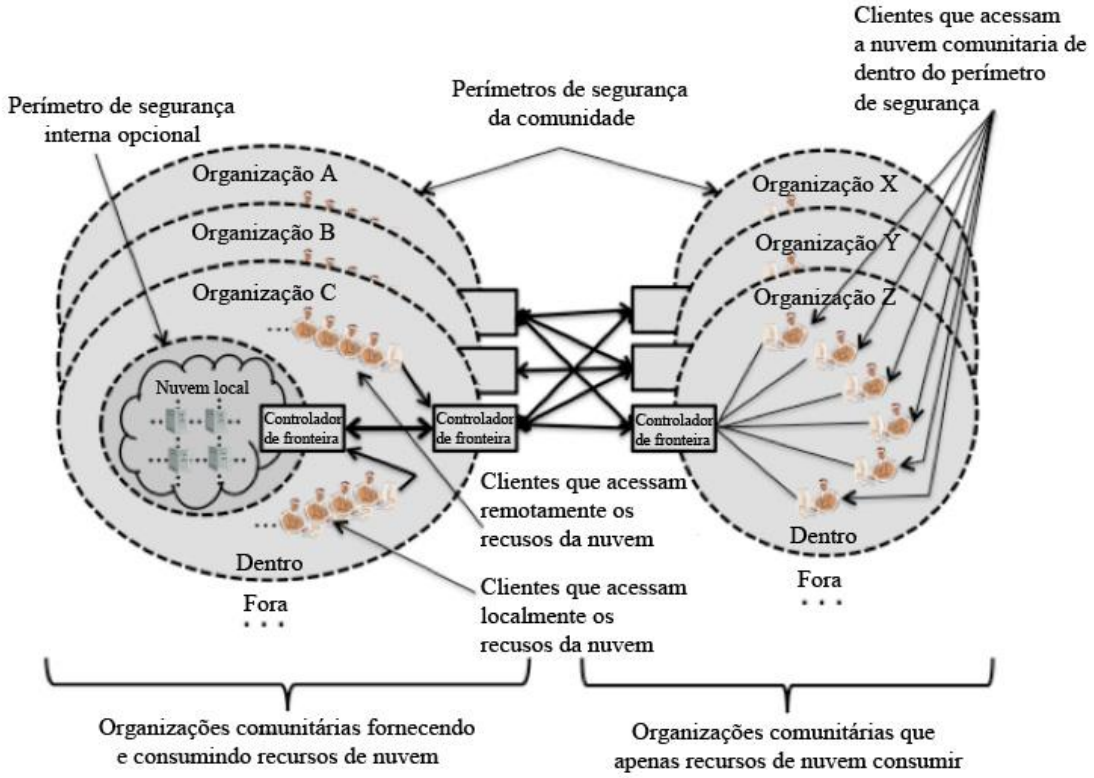
Uma infraestrutura compartilhada por muitas organizações e suporta uma comunidade que partilham as mesmas as mesmas preocupações, como as missões, os requisitos de segurança, política e considerações de conformidade (BADGER, 2011).

Sendo usada para controlar os dados e assim disponibilizar os acessos aos servidores, unindo os conceitos da nuvem privada, com as da nuvem pública e com a nuvem comunitária, e assim criando uma nuvem com foco ao publico e a empresas, mas garantindo a segurança dos dados (NAITO, 2010).

Tendo como exemplo algumas aplicações que disponibilizam o modelo de nuvem privada (BASANT, 2011):

- *Google Apps for Government*;
- *Microsoft Government Community Cloud*.

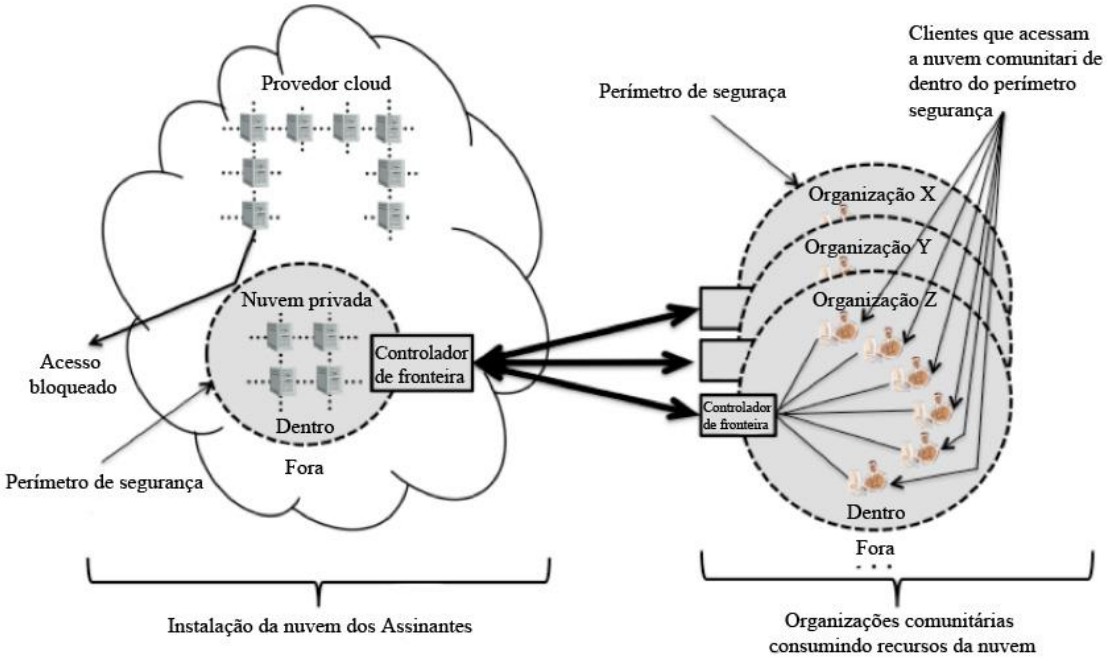
Figura 8 – Nuvem Comunitária On-Site.



Fonte: (BADGER, 2011, Adaptado pelos autores).

A Figura 8 ilustra o conceito da Nuvem Comunitária *On-Site* onde um conjunto de organizações particulares pode fornecer ou utilizar serviços em nuvem. Sendo necessário pelo menos um membro da comunidade fornecer serviços de nuvem para nuvem para ser funcional. E cada organização implementa linhas de segurança, assim as mesmas então conectadas através de links entre os controladores de fronteiras. A figura ilustra a esquerda os membros que fornecem serviços em nuvem e a direita os que somente utilização os serviços (BADGER, 2011).

Figura 9 – Nuvem Comunitária Terceirizada.



Fonte: (BADGER, 2011, Adaptado pelos autores).

A Figura 9 ilustra o conceito de nuvem comunitária terceirizada, de modo que o conjunto organizações participantes usufruem os serviços da nuvem. Sendo muito semelhante ao cenário da nuvem privada terceirizada de modo que os servidores são gerenciados por provedores que implementam um perímetro de segurança. (BADGER, 2011).

2.3.5.4 Nuvens híbridas (*Hybrid Cloud*)

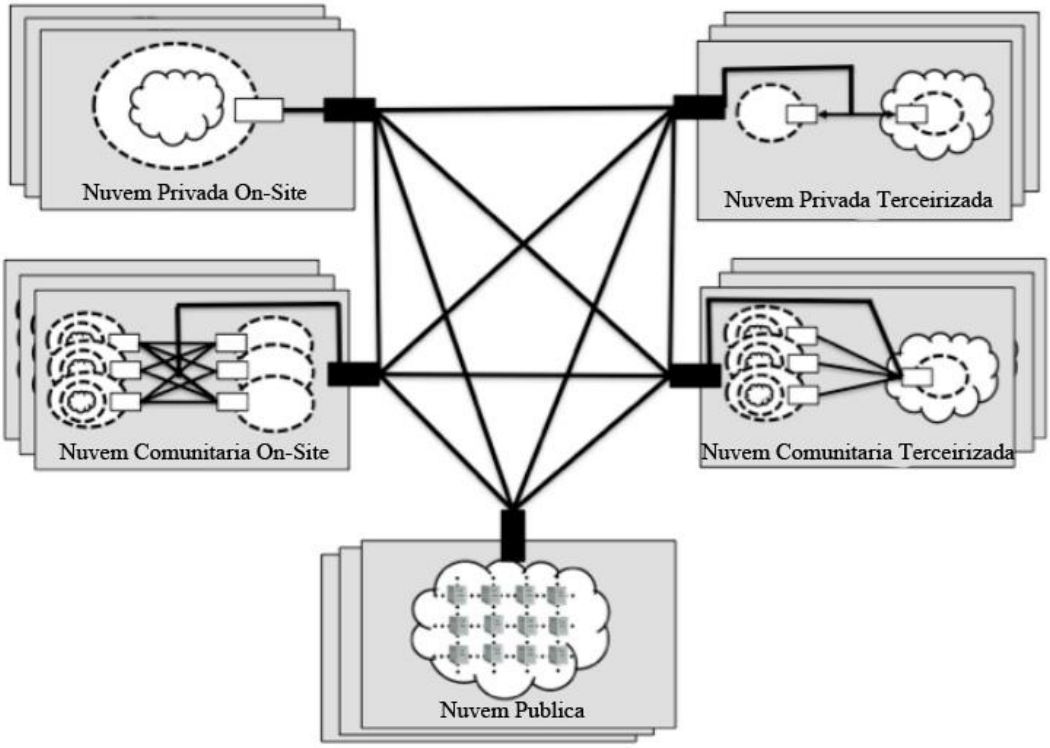
Segundo Sousa (2009) a nuvem hibrida é a "[...] composição de duas ou mais nuvens, que podem ser privadas, comunidade ou pública e que permanecem como entidades únicas, ligadas por uma tecnologia padronizada ou proprietária que permite a portabilidade de dados e aplicações".

Tendo como exemplo algumas aplicações que disponibilizam o modelo de nuvem privada (BASANT, 2011):

- Windows Azure

- VMware vCloud

Figura 10 – Nuvem Híbrida.



Fonte: (BADGER, 2011, Adaptado pelos autores)

A Figura 10 ilustra como uma nuvem híbrida pode ser formada através dos tipos de nuvens. Podendo ser formadas por vários pontos de acesso para as nuvens que à compõe. (BADGER, 2011)

2.3.5.5 Intercloud

É formado por nuvens interligadas de modo a oferecer um ambiente universal em computação em nuvens em padrões abertos. De acordo com NAITO (2010) "[...] se referindo a "nuvem de nuvens", ou união de diversas nuvens para compartilhamento de recursos entre os servidores, tornando assim quase infinita a quantidade de recursos disponíveis".

2.3.6 Implicações na TI

Mesmo as tecnologias que surgem para facilitar, têm suas implicações, as implicações surgem pela maneira de comunicação e pelas mudanças na interatividade dos negócios, de modo que estes serviços necessitam de uma internet super-rápida para que suas funcionalidades estejam à frente de seu tempo e o principal a frente de seus concorrentes. (MARTINS, 2010).

Segundo Carneiro (2011) “O maior desafio a ser enfrentado pela Computação nas Nuvens é a segurança. Para entender os potenciais riscos de segurança, as empresas devem fazer uma avaliação completa de um serviço de nuvem – começando com a rede, checando as operações do fornecedor e desenvolvendo o aplicativo em nuvem.”.

Outro ponto seria a disponibilidade destes serviços, de modo que as informações estão disponíveis nas nuvens e são influenciadas pelos atrasos e pelas indisponibilidades causadas pela internet. Para evitar os ambientes de computação em nuvem devem prover alta disponibilidade, e assim esses podem usar técnicas de balanceamento de carga dinâmico e composição de nuvens de forma a atender as necessidades dos usuários. Tendo como exemplo, implantação de uma aplicação em nuvens diferentes, caso uma nuvem falhe a outra continua disponibilizando a aplicação. (SOUSA, 2009).

2.3.7 Modelo de Arquitetura

Nesta seção são apresentados alguns modelos de arquitetura para viabilizar uma solução de comunicação nas nuvens. A divisão entre os modelos aqui apresentados, levam em consideração características do tipo, tecnologia base utilizada e contexto do serviço em questão.

Os modelos apresentados são: Modelo de *grid computing*, modelo de *cluster* (agrupamento) e modelo de armazenamento.

2.3.7.1 Modelo de *Grid Computing*

O conceito de grid começou por volta da década de 90 com o intuito de resolver os problemas computacionais, de modo que os recursos computacionais de alta escala eram muito caros e difíceis de adquirir. Para resolver o problema foi utilizada uma rede com máquinas interligadas compartilhando os recursos computacionais, assim simulando um computador de grande escala. (FOSTER, 2008).

Tendo como objetivo conciliar tecnologias heterogenias, obtendo sistemas robustos, dinâmicos e escaláveis, de modo a compartilhar processamento, espaços de armazenamento, dados, aplicativos e entre outros. (MARTINS, 2010).

Segundo Tagliri (2006)

> O Grid Computacional pode ser entendido como uma rede transparente, onde o usuário se conecta para obter benefícios computacionais sob demanda. Esses recursos ficam previamente cadastrados pelas organizações virtuais como Universidades, Centros de Pesquisa e outros, sendo que para utilizá-los há um custo definido pelas mesmas.
>
> Mas o Grid pode também ser formado dentro de uma organização, e um bom exemplo é o banco de dados Oracle 10g. Ele utiliza a Computação em Grade para dividir entre os computadores da rede, a carga de processamento que pode ser gerada. Além disso, caso ocorra falha entre um dos computadores, automaticamente a carga é distribuída entre os outros restantes na rede. Nesse caso, ele calcula como será distribuído o processamento entre os computadores que estão em funcionamento e realiza a divisão para obtenção de um melhor aproveitamento da rede.

Segundo Foster (1999, apud Tagliri, 2006) a arquitetura de um *grid* computacional é composta por cinco camadas:

a) Aplicação: composto pelas aplicações dos usuários que trabalham no ambiente da organização virtual.

b) Coletivo: camada responsável pela definição dos protocolos e serviços utilizados para atuar nas interações entre coleções de recursos.

c) Recursos: camada onde se encontra a definição dos protocolos utilizados, e são eles: Protocolos de Informação que são usados para obtenção de informações sobre a estrutura e o estado dos recursos compartilhados e Protocolos de Gerenciamento que negociam acesso a recursos compartilhados.

d) Conectividade: camada onde são definidos os protocolos básicos de comunicação e autenticação do *Grids*

e) Ambiente: nesta camada existe a interface para controle local dos recursos disponibilizados

Para resumir o que seria um *grid*, tem como objetivo disponibilizar recursos e serviços de uma forma geograficamente distribuída (DANTAS, 2005).

2.3.7.2 Modelo de *Cluster*

O modelo de *cluster* surgiu em 1960, quando a IBM interligou os mainframes e obteve um ganho no processamento no paralelismo. E em 1980 com as redes de alta velocidade, microcomputadores mais potentes e ferramentas padronizadas para computação distribuída de alto desempenho se teve um avanço na tecnologia de cluster. (CONTI, 2008).

Sendo formado por um conjunto de computadores com um sistema operacional classificado com um sistema distribuído, muitas vezes construído a partir de computadores convencionais, e são interligados através de uma rede de forma que a comunicação através do sistema. (MARTINS, 2010).

Segundo Bacellar (2011) o sistema operacional deve garantir os seguintes funcionamentos:

a) Controle de recursos: Nessa tarefa o sistema operacional realiza um controle de entrada e saída de cada nó do *cluster* envolvido na realização da tarefa, e assim, garantir a consistência dos dados executados.

b) Monitoração: Tarefa responsável por verificar a disponibilidade de cada nó do *cluster*, assim como a quantidade de carga que cada nó poderá receber.

c) Contabilidade: Essa tarefa funciona como um medidor de desempenho do cluster seja ela, para calcular custos da operação ou até mesmo medir desempenho.

d) Consulta: Nesta tarefa o sistema operacional armazena em fila de prioridades todos os processos que serão executados, levando em consideração as tarefas executadas por diferentes níveis de usuários.

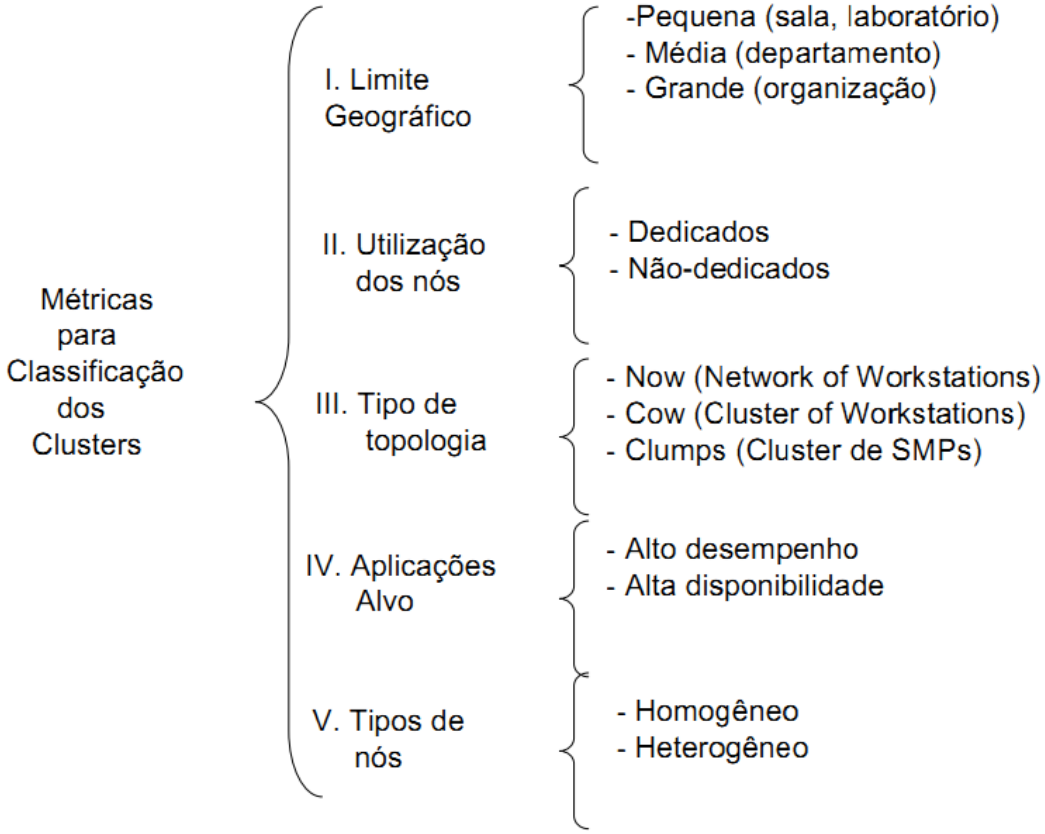
e) Planificação: Funciona como um planejador, e é através dele que o sistema operacional organiza da melhor maneira quais tarefas serão distribuídas entre os nós do cluster.

Conforme Bortolin (2005)

> As tecnologias de clusters permitem às organizações incrementar o poder de processamento, utilizando hardware e software que são padrão de mercado; podendo ser adquiridos a custo relativamente baixo. Isso assegura expansibilidade e uma boa maneira de aumentar o poder computacional, preservando o investimento existente.
>
> O desempenho das aplicações também melhora com suporte em um ambiente escalável de software. Outro beneficio da clusterização é redundância permitindo que um computador backup responda pelas tarefas do computador que falhou ou se tornou inoperante, designado como failover.

Na Figura 11 mostra como Dantas (2005) apresenta as métricas para a classificação dos *clusters.*

Figura 11 – Métricas para Classificação dos *clusters*



Fonte: (DANTAS, 2005).

E para resumir o que seria um *cluster*, é uma configuração de *software* e *hardware* tendo como o objetivo de solucionar os problemas de uma organização. (DANTAS, 2005).

2.3.7.3 Modelo de Armazenamento

Os arquivos são usados para abstrair os dados de um sistema computacional, de modo que é necessária uma estrutura chamada de sistema de arquivo, que permite que os arquivos sejam acessados, modificados e que possam criar novos arquivos. (MARTINS, 2010).

Soares (2011) relata que um sistema de arquivo distribuído (SAD) é um sistema no qual os arquivos armazenados são espelhados em diversos hardwares através de uma rede. Tendo semelhanças como aos sistemas de arquivo centralizado, podendo manipular arquivos, controlar redundância, consistência. Tendo que prover transparências nos contextos citados abaixo:

- De acesso: aplicações que acessam os arquivos do SAD não devem estar cientes da localização física deles.
- De localização: todas as aplicações devem ter sempre a mesma visão do espaço de arquivos.
- De mobilidade: com a movimentação dos arquivos, nem programas do cliente e nem tabelas de administração precisam ser modificadas, de modo a refletir essa movimentação.
- De desempenho: programas clientes devem executar satisfatoriamente, independente de variação de carga do serviço de arquivos.
- De escalabilidade: o serviço pode ser expandido por crescimento horizontal, e não vertical, de modo a se adequar à carga demandada e à capacidade da rede disponível.

Tendo como um bom exemplo deste modelo o *dropbox* (www.dropbox.com), a onde os dados são compartilhados entre todos os computadores que tem permissão.

2.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Neste capitulo, foram expostos os conceitos básicos para o desenvolvimento do trabalho através de um levantamento bibliográfico baseado em livros e artigos publicados em periódicos e congressos, nacionais e internacionais.

Pode-se perceber que a computação nas nuvens faz com que o acesso a novos recursos computacionais seja feita de forma provisionada e também que os preços estão cada vez mais atrativos na medida em que esse modelo vem ganhando força no mercado.

As modalidades de computação nas nuvens estudadas foram, SaaS que disponibiliza o *software* como serviço, o PaaS que disponibiliza a plataforma como serviço e o IaaS que disponibiliza a infraestrutura como serviço.

O estudo do SOA contribuiu para o entendimento deste novo paradigma orientado a serviço. Observou-se que a relação de SOA com SaaS está relacionado com a modelagem e disponibilidade do sistemas que é desenvolvido como serviço que é disponibilizado a partir de uma modalidade.

3 METODOLOGIA

Segundo Barros e Lehfeld (2000, p. 1) os mesmos afirmam que metodologia:

> Consiste em estudar e avaliar os vários métodos disponíveis, identificando suas limitações ou não em nível das implicações de suas utilizações. A metodologia, em um nível aplicado, examina e avalia as técnicas de pesquisa, bem como a geração ou verificação de novos métodos que conduzem à capacitação e processamento de informações com vistas à resolução de problemas de investigação.

Menezes e Silva (2005, p. 25) afirmam que metodologia científica é "[...] o conjunto de processo ou operações mentais que se devem empregar na investigação. É a linha de raciocínio adotada no processo de pesquisa".

Este capítulo aborda a metodologia utilizada para apoiar o desenvolvimento do Trabalho de Conclusão de Curso (TCC), tendo como finalidade, mostrar o tipo de pesquisa que é realizado nesta proposta, as etapas metodológicas, o cronograma, a proposta de solução e as principais delimitações.

3.1 TIPOS DE PESQUISA

Menezes e Silva (2005, p.19) definem que "Pesquisar significa, de forma bem simples, procurar respostas para indagações propostas". Os mesmos autores complementam afirmando que:

> Pesquisa é um conjunto de ações, propostas para encontrar a solução de um problema, que têm por base procedimentos racionais e sistemáticos. A pesquisa é realizada quando se tem um problema e não se têm informações para solucioná-lo.

Existem diversas formas de se definir os tipos de pesquisas, dentre elas tem-se: a pesquisa básica, aplicada, quantitativa, qualitativa, exploratória, descritiva, explicativa, bibliográfica documental, experimental, levantamento, estudo de caso, expost-facto, ação e participante (MENEZES E SILVA, 2005, p. 20).

3.1.1 Pesquisa Bibliográfica

Segundo Menezes e Silva (2005, p. 21, apud GIL, 1991) a pesquisa bibliográfica é definida como: “quando elaborada a partir de material já públicado, constituído principalmente de livros, artigos de periódicos e atualmente com material disponibilizado na Internet”.

Conforme Barros (2000) a pesquisa bibliográfica é “a que se efetua tentando-se resolver um problema ou adquirir conhecimento a partir do emprego predominante de informações advindas de material gráfico, sonoro e informatizado”.

Para Ruiz (1982) a mesma se “consiste no exame de material já públicado, para levantamento e análise do que já se produziu sobre determinado assunto que se assumiu como tema de pesquisa científica”.

A pesquisa bibliográfica visa fundamentar este trabalho a partir de referencias relacionado à comunicação nas nuvens, com o objetivo de acrescentar conhecimento baseado em evidências tecnológicas e científicas.

3.1.2 Pesquisa Aplicada

Para Menezes e Silva (2005, p. 20) a pesquisa aplicada “objetiva gerar conhecimentos para aplicação prática e dirigidos à solução de problemas específicos. Envolve verdades e interesses locais”.

Segundo Danton (2002, p. 10) é a “busca de solução para problemas concretos e imediatos. Muitas vezes pesquisas puras revelam grande importância em nossa vida. É o caso da eletricidade. Quando os primeiros cientistas começaram a pesquisá-la, o único objetivo era a curiosidade”.

Este tipo de pesquisa foi realizado, devido ao fato da grande procura por esse novo modelo computacional, neste trabalho é mostrado as vantagens e desvantagens em

relação à maneira tradicional com o esse novo modelo e quais os atuais modelos disponíveis no mercado para esse desenvolvimento nas nuvens.

3.1.2.1 Pesquisa Aplicada

É o método pelo qual conseguimos unir conhecimentos adquiridos, através da revisão bibliográfica com os conhecimentos acadêmicos adquiridos ao longo do curso para a elaboração do objetivo proposto deste trabalho.

3.1.3 Estudo de Caso

Este trabalho consiste em uma pesquisa do tipo Estudo de Caso que se caracteriza como sendo um "[...] estudo profundo e exaustivo de um ou de poucos objetos, de maneira que permita o seu amplo e detalhado conhecimento, tarefa práticamente impossível mediante os outros delineamentos já considerados" (GIL, 2002, p. 54). Sendo assim, "o alvo são as características que o caso tem de único, singular ou particular. Mesmo que existam casos similares, um caso é distinto e, por isso, causa interesse próprio" (RAUEN, 2002, p. 210).

Além de definir o universo de pesquisa, o estudo de caso apresenta outras vantagens, como o estímulo a novas descobertas, que abre a possibilidade de o pesquisador se deparar com fatos interessantes que não haviam sido previstos no plano inicial. (GIL, 2002).

3.1.3.1 Estudo de Caso

É o método pelo qual essa pesquisa visa coletar, compreender e descrever os resultados obtidos através desta situação específica que é composta por este trabalho.

3.2 TIPOS DE PESQUISAS QUE SE APLICAM AO TRABALHO

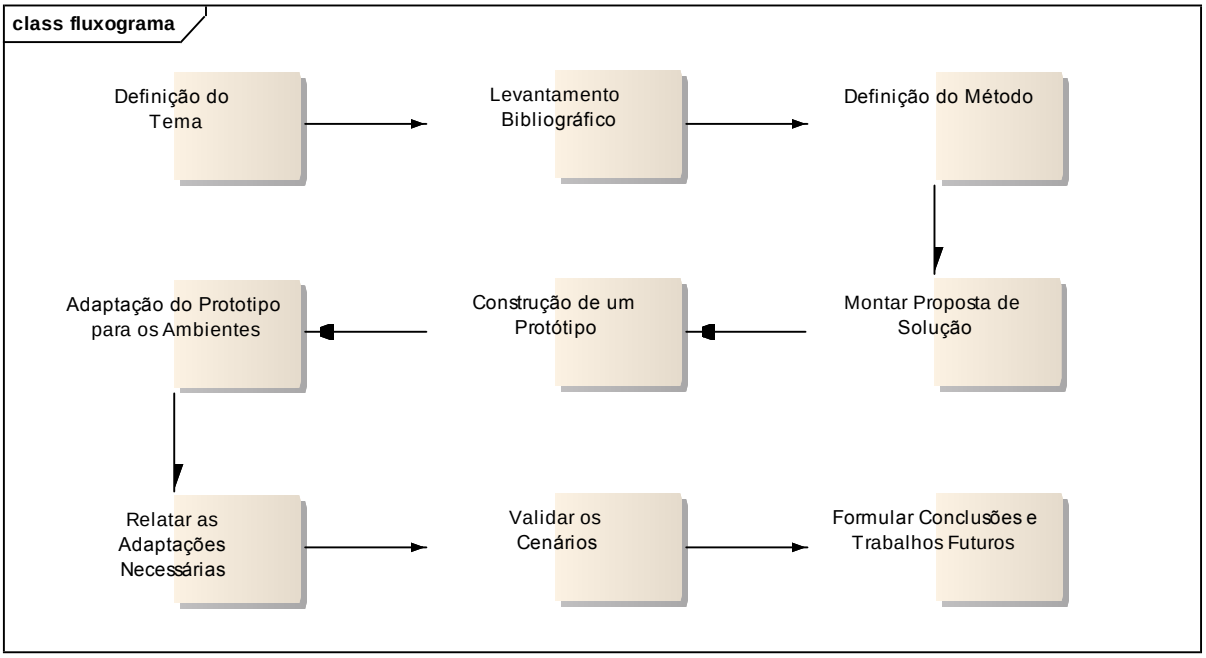
Dos tipos de pesquisas supramencionados, este projeto se enquadra nos seguintes tipos:

3.3 ETAPAS

Este tópico visa abordar todas as etapas necessárias para a elaboração e execução do projeto, dentre elas estão: a pesquisa bibliográfica, aplicada e de campo, o levantamento de requisitos, a modelagem do problema com foco na solução, o desenvolvimento da aplicação, testes e validações e a implantação da aplicação em algumas nuvens atualmente disponíveis no mercado, com o objetivo de identificar as mudanças necessárias para cada plataforma.

As etapas estão ilustradas conforme o fluxograma da Figura 12.

Figura 12 – Fluxograma das etapas do trabalho.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 12 mostra um fluxograma apresentas as etapas para elaboração do trabalho, e para um melhor entendimento serão esclarecidas as etapas abaixo:

- Definição do Tema: Foi escolhido junto ao orientador;
- Levantamento Bibliográfico: São levantadas as informações para obter uma base introdutória para entendimento do trabalho;
- Definição do Método: São definidas as formas de pesquisas para o levantamento dos dados necessários para a elaboração do trabalho;
- Montar Proposta de Solução: É elaborada uma proposta que se adapte ao tema escolhido;
- Construção de um Protótipo: Como o próprio nome diz, será construído um protótipo;
- Adaptação do Protótipo para os Ambientes: São feitas modificações no protótipo, para que o mesmo possa ser aplicado em outros ambientes;
- Relatar as Adaptações Necessárias: Relatar o que foi necessário para realizar as adaptações no protótipo para todos os cenários;
- Validar os Cenários: São executados estudos para avaliação dos cenários a partir da implantação do protótipo nas modalidades PaaS e IaaS, a fim de verificar se mesmo depois da adaptação o protótipo ainda atende as

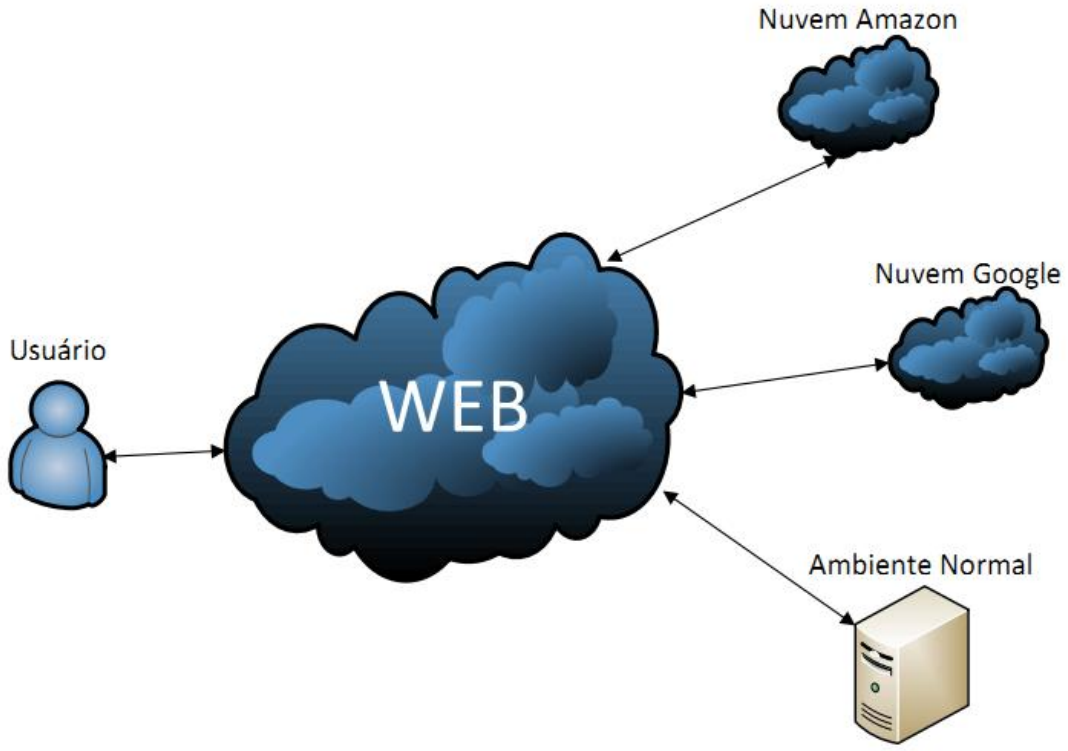
características da comunicação nas nuvens.

- Formular Conclusões e Trabalhos Futuros: São elaboradas as conclusões gerais e específicas, e assim como a descrição de trabalhos futuros.

3.4 PROPOSTA DE SOLUÇÃO

O trabalho tem como objetivo a comparação entre os modelos de implementação em nuvens, sendo realizado os estudos a partir da modelo IAAS disponibilizada pela Amazon, pelo modelo PAAS disponibilizada pela Google e também em um ambiente local.

Figura 13 – Esquema da Pesquisa.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A Figura 13 ilustra o ambiente onde serão realizados os estudos propostos. Primeiramente foi desenvolvido o sistema proposto para um ambiente normal e em seguida será feita a migração para os outros ambientes propostos com o propósito de realizar uma

comparação entre a nuvem disponibilizada pela Amazon que adota o os conceitos IAAS, o da Google que adota os conceitos PAAS e em um ambiente tradicional.

3.5 DELIMITAÇÕES

Devido à alta complexidade e ao curto tempo para o desenvolvimento da aplicação, foram definidas algumas delimitações para esse projeto.

Delimitações deste projeto:

- O trabalho visa o desenvolvimento de uma aplicação simples, com a finalidade de apresentar as alterações necessárias para o funcionamento na nuvem da Amazon e na do Google;
- Não é o objetivo deste projeto tratar aspectos relacionados a segurança nas nuvens;
- Não é o objetivo deste projeto apresentar indicativos de como desenvolver um serviço para a computação nas nuvens;
- Não é objetivo definir qual o melhor *hardware* para o desenvolvimento desta aplicação;
- Não é objetivo definir em qual nuvem é mais vantajoso manter este aplicativo.
- Não é objetivo deste projeto definir que tipos de serviços *Cloud Computing* pode oferecer.

4 PROJETO DE SOLUÇÃO

Nesta seção serão apresentadas as definições de técnica e metodologia, sendo abordado um pouco sobre UML, Orientação a objetos, o modelo Iconix e em seguida a apresentação do estudo de caso proposto para que atenda tanto a modalidade IaaS quanto a modalidade PaaS.

4.1 DEFINIÇÃO DE TÉCNICA E METODOLOGIA

Nesta seção são mostradas as técnicas e as metodologias adotadas na modelagem. São elas: UML, ICONIX e orientação a objeto.

4.1.1 Unified modeling language (UML)

O UML é uma linguagem de modelagem designada para especificar, visualizar, construir e documentar um sistema. Podendo ser usado em todos os processos do ciclo de desenvolvimento projeto. (FURLAN, 1998).

O surgimento o UML se deu pela unificação de muitas linguagens gráficas de modelagem orientadas a objetos que floresceram entro o fim dos anos oitenta e noventa, e assim com o intuito de resolver a torre de Babel que se tinha surgiu em 1997 o UML. (Fowler, 2005).

Sendo considerado um padrão aberto, mas controlado pelo OMG (Object Management Group), um consórcio aberto de empresas. Criado para estabelecer padrões que suportassem interoperabilidade, especificamente a de sistemas orientados a objetos. (Fowler, 2005).

4.1.2 ICONIX

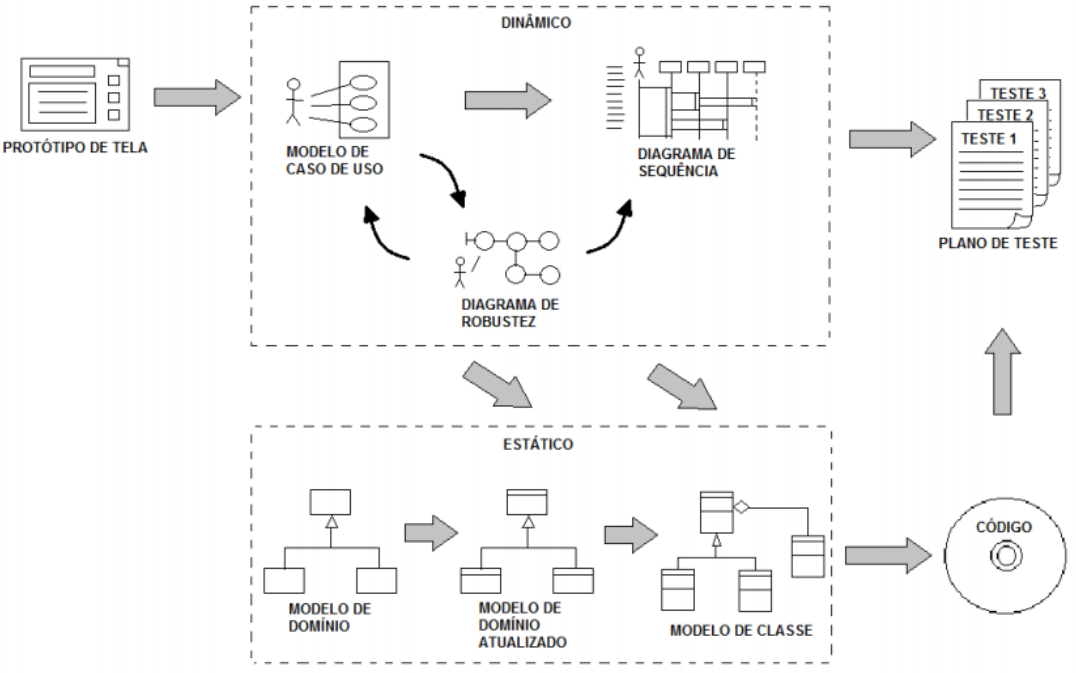
O ICONIX é uma metodologia de desenvolvimento de software desenvolvida pela empresa ICONIX Software Engineering. É uma metodologia muito prática, conduzida por casos de utilização iterativo e incremental, como o *Rational Unified Process* (RUP), mas sem a sua complexidade. E também, é relativamente pequeno e simples, como o *Extreme Programming* (XP), mas sem deixar de lado as tarefas de análise e de desenho. (SILVA, A. et al, 2001).

Segundo Silva, G et. al. (2007) os principais diagramas do processo ICONIX são os seguintes:

- Modelo de Domínio
- Modelo de Caso de Uso
- Diagrama de Robustez
- Diagrama de Sequência
- Diagrama de Classe

A figura 14 mostra a visão geral do desenvolvimento de um software utilizando a metodologia ICONIX

Figura 14 – Visão geral do ICONIX.



Fonte: (ADAPTADO PELOS AUTORES, ROSENBERG et al, 2005).

De acordo com Rosenberg et al (TRADUÇÃO NOSSA, 2005) o ICONIX pode ser dividido nos seguintes passos:

- Passo um: Identificar os objetos do mundo real de domínio através do modelo de domínio;
- Passo dois: Definir os requisitos comportamentais através dos casos de uso;
- Passo três: Realizar diagrama de robustez para eliminar os casos de uso iguais e identificar falhas no modelo de domínio;
- Passo quatro: Definir o comportamento das classes e objetos através do diagrama de sequência;
- Passo cinco: Concluir o modelo estático apartir do diagrama de classes;
- Passo seis: Gerar o código;
- Passo sete: Executar o sistema e fazer testes para a aceitação do usuário.

Desta forma foi utilizada esta metodologia para este trabalho por se tratar de uma metodologia menos burocrática em relação as outras metodologias.

4.1.3 Orientação a objeto (OO)

A orientação a objetos veio para solucionar as dificuldades ocasionadas pela programação estruturada. Com a orientação a objetos se divide o código em vários blocos pequenos chamados de objetos, assim pode se reutilizar os códigos. Tendo como princípios: (BOGGS et al, 2002).

- Encapsulamento - Esconder os detalhes da implementação de um objeto.
- Herança - é o mecanismo pelo qual uma classe pode estender outra classe, aproveitando seus comportamentos e variáveis possíveis.
- Polimorfismo - é ter varias formas ou implementações de uma determinada função.

De acordo com Meyer et al (1996, p.8) a orientação a objetos apresenta as principais vantagens em um desenvolvimento:

- Abstração de dados: não é preciso ser um conhecedor dos detalhes de uma classe especifica para que possa ser usada em outra classe;
- Compatibilidade: É a facilidade com que o programa pode ser combinado com outros.
- Flexibilidade: se refere à capacidade alterar a funcionalidade presente, sem consequências imprevistas sobre o conjunto da estrutura;
- Reutilização: Permitir que o código possa ser utilizado mais de uma vez;
- Extensibilidade: se refere à capacidade ampliar a funcionalidade presente, sem consequências imprevistas sobre o conjunto da estrutura;
- Manutenção: Através da divisão do código entre as classes e com a facilidade de entendimento, a manutenção se torna mais fácil.

4.2 MODELAGEM DO SISTEMA PROPOSTO

Nesta seção será apresentada a modelagem do sistema proposto, na seguinte ordem: atores, requistos, protótipos de tela, casos de uso primário, modelo de domínio, diagrama de robustez e diagrama de sequência.

4.2.1 ATORES

Os atores são uma entidade externa (fora do sistema) que interage com o sistema, também podendo participar de mais de um papel no sistema. Cada ator deve possuir um nome, cujo terá relação direta com a sua função, possuirá uma descrição que definirá o que ele faz e com quem ele interage. (ALMEIDA et al, 2001).

A figura 15 ilustra o ator que interage com o sistema proposto.

Figura 15 – Ator.

Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

4.2.2 REQUISITOS

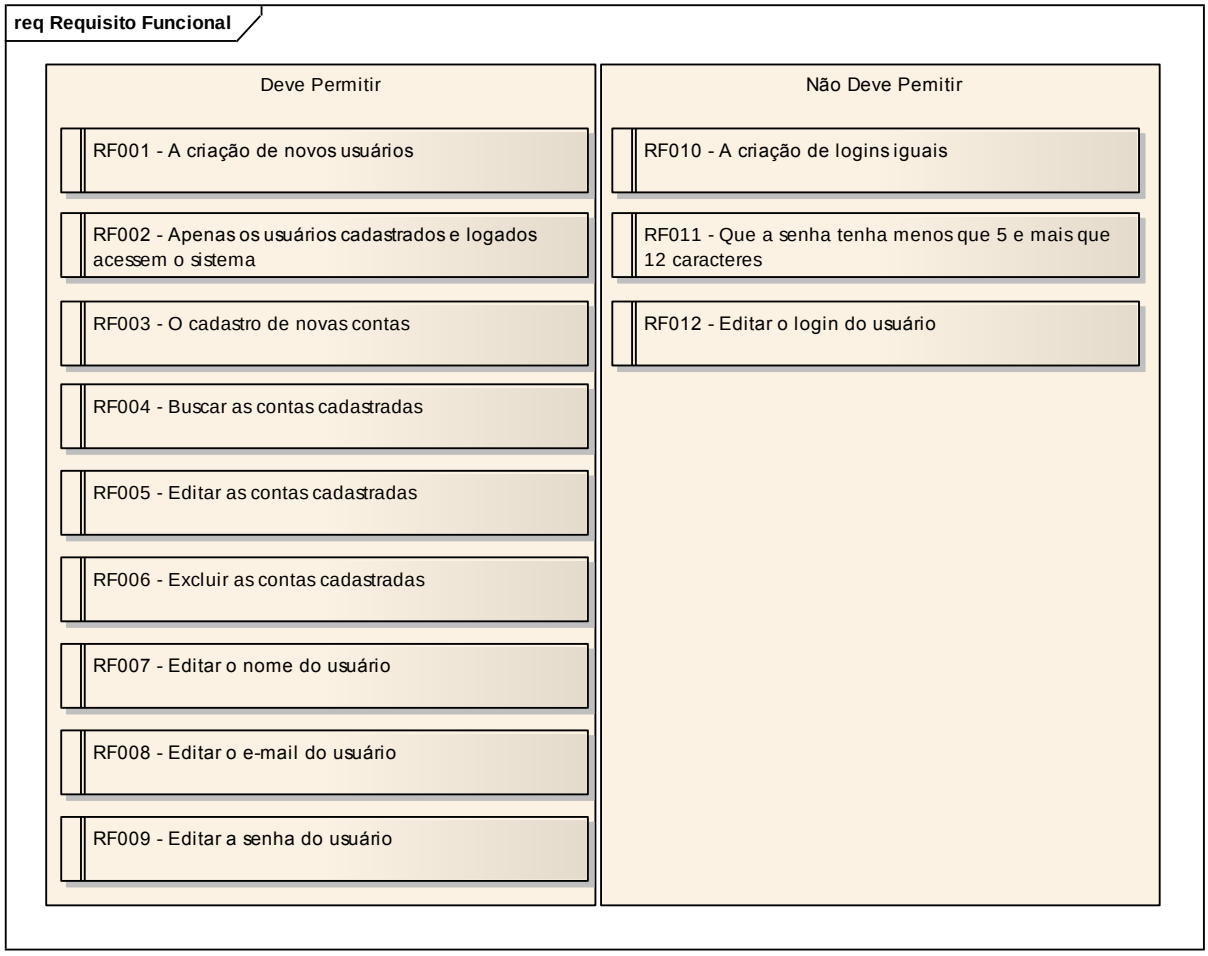
De acordo com Engholm Jr. (2010) “Os requisitos definem as expectativas e necessidades dos envolvidos no projeto, podendo ser divididos em requisitos funcionais e não funcionais”.

4.2.2.1 REQUISITOS FUNCIONAIS

Os requisitos funcionais são funcionalidades propostas pelo sistema, de modo a ser usados para resolver problemas e atender a todas as necessidades funcionais dos usuários do sistema. (Longo et al., 2005).

A figura 16 ilustra os requisitos funcionais do sistema proposto.

Figura 16 – Requisitos funcionais.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

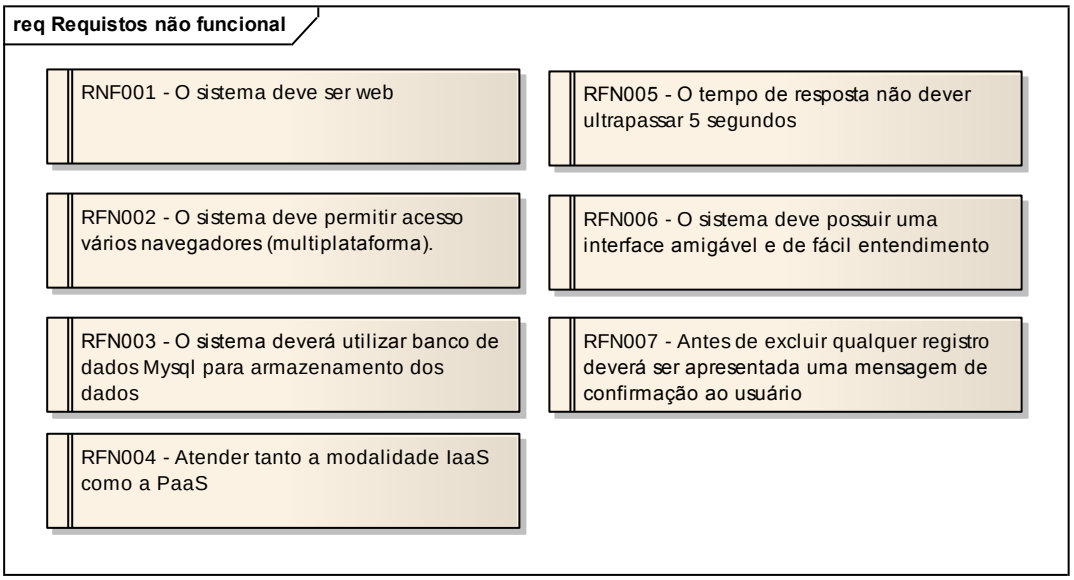
4.2.2.2 REQUISITOS NÃO FUNCIONAIS

Geralmente ligados à qualidade do produto, tendo como exemplo, robustez, segurança ou integridade do sistema em questão. As qualidades do sistema têm dois sub-ramos (Longo et al., 2005):

- Dependência positiva: Quando da presença de um o outro é mais facilmente incorporado.
- Dependência negativa: São conflitantes entre si, a presença ou existência de um compromete o outro ou diminui seu grau de atuação.

A figura 17 ilustra os requisitos não funcionais do sistema proposto.

Figura 17 – Requisitos não funcionais.



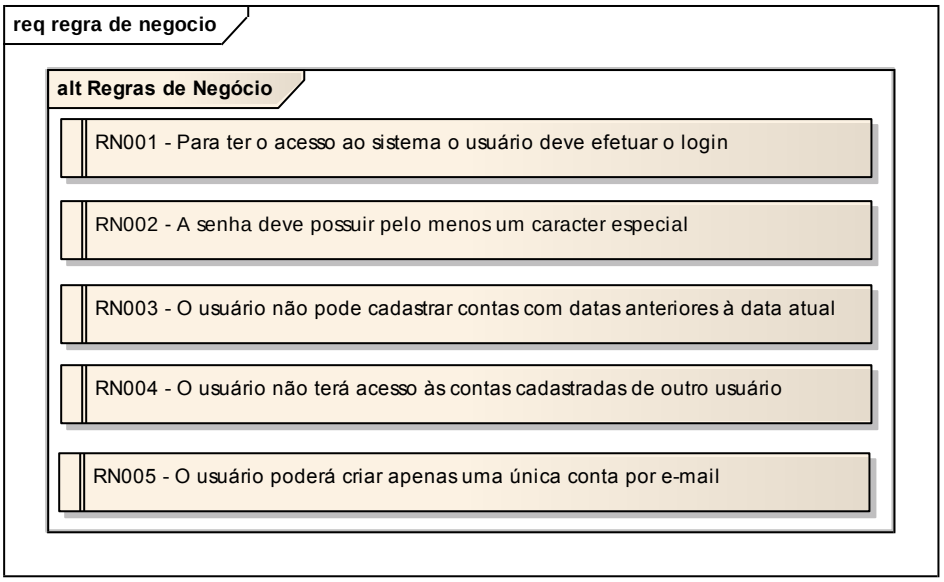
Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

4.2.2.3 REGRAS DE NEGÓCIO

As regras definem como o processo deve ser conduzido, podem definir como os recursos devem ser estruturados e relacionados uns com os outros.

A figura 18 ilustra as regras de negócio do sistema proposto.

Figura 18 – Regra de Negócio



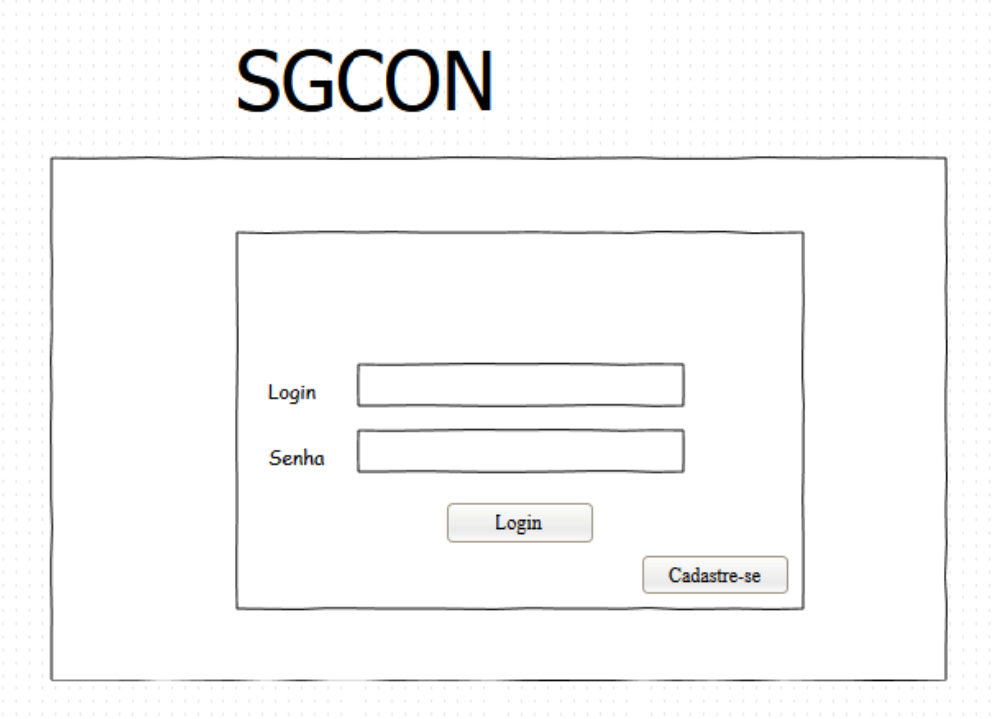
Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

4.2.3 PROTÓTIPOS DE TELA

Nesta seção serão apresentados todos os protótipos de tela do sistema, facilitando a visualização de cada ação no sistema.

A figura 19 apresenta a tela inicial, onde o usuário irá realizar o login no sistema, além de ter a opção de se cadastrar no sistema.

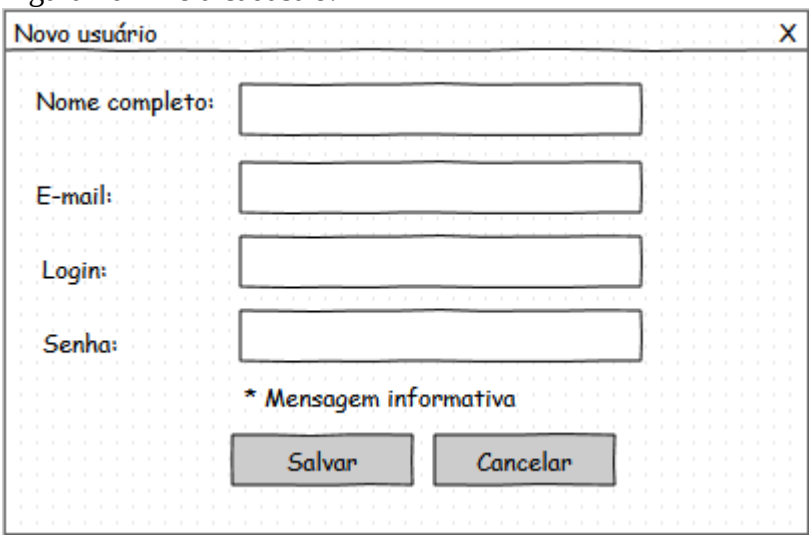
Figura 19 – Tela login.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 20 exibe a tela de cadastro, nesta tela o usuário obrigatoriamente tem que preencher todos os campos, caso não seja preenchido, será apresentada uma mensagem ao lado de cada campo que deixou de ser preenchido. Ao final do processo de preenchimento dos campos e da ação de salvar será apresentada uma mensagem no espaço mensagem informativa, informando mensagens de sucesso ou erro.

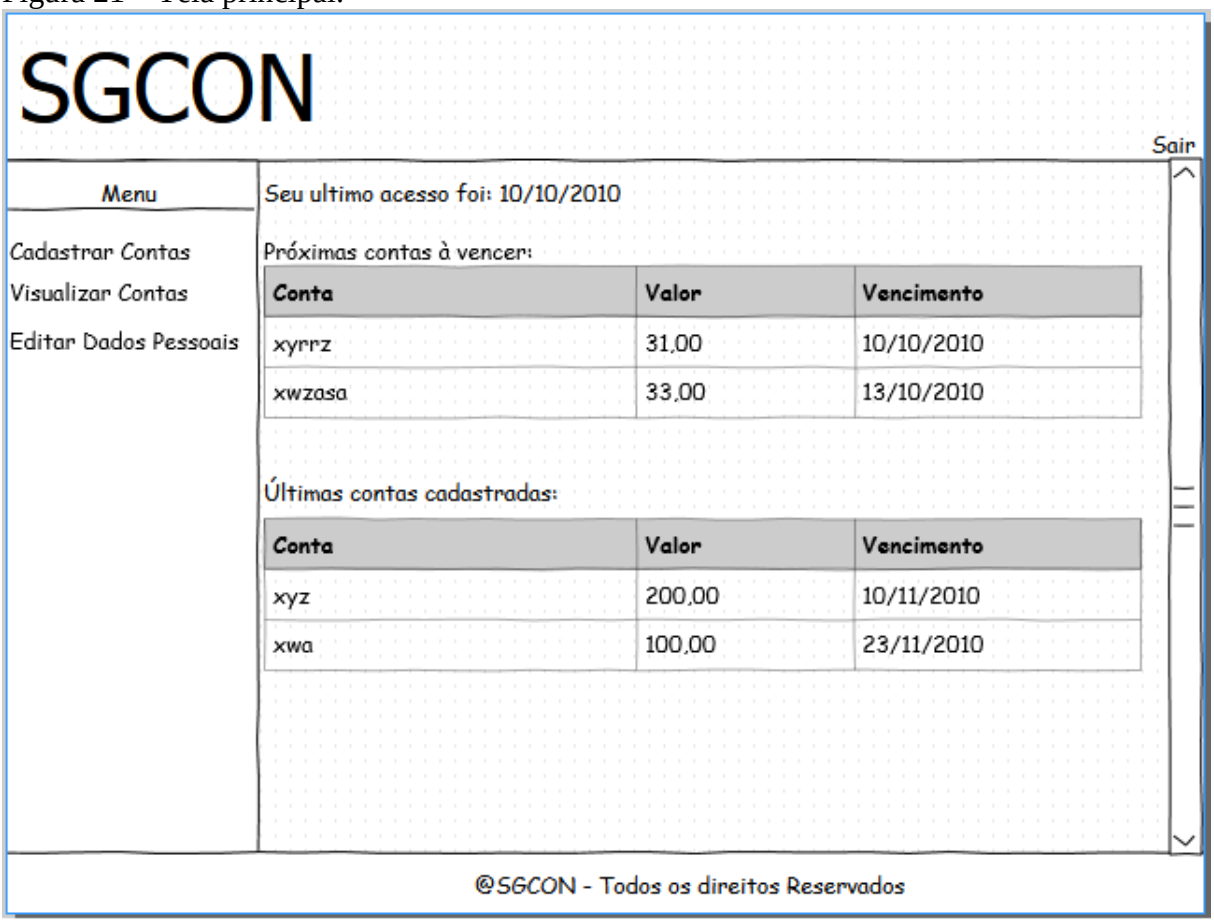
Figura 20 – Tela cadastro.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 21 ilustra a tela principal, esta tela o usuário tem uma rápida visualização das próximas contas que estão vencendo do seu último acesso e das suas últimas duas contas cadastradas. Ao lado esquerdo da tela possui um menu para a execução de determinadas ações e no parte superior direita tem um botão para efetuar o *logoff* no sistema.

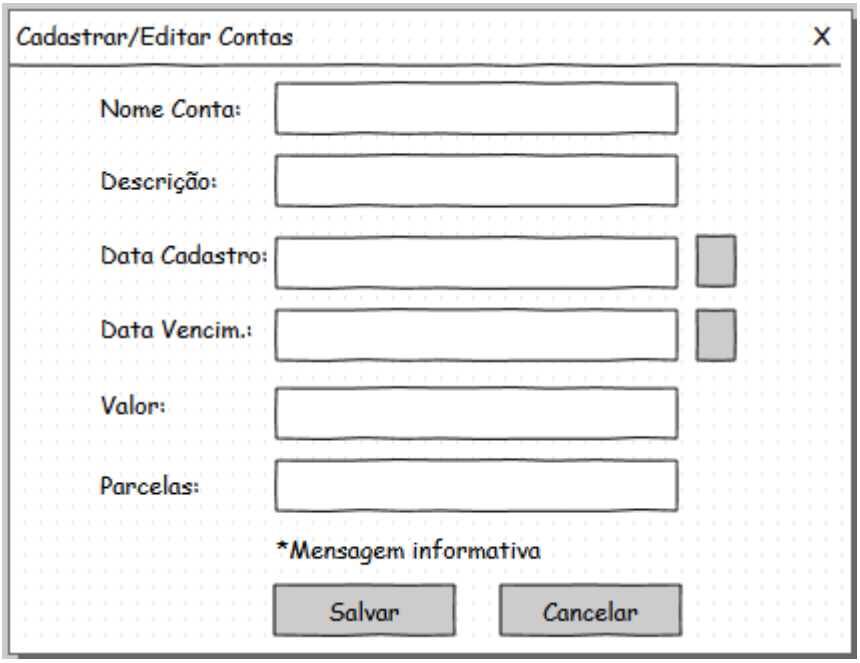
Figura 21 – Tela principal.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 22 exibe a tela que será utilizada tanto para cadastro de uma nova conta, quanto para a edição de uma conta previamente cadastrada e selecionada na tela de visualização de contas. Ao final da ação de salvar será exibida uma mensagem no campo mensagem informativa.

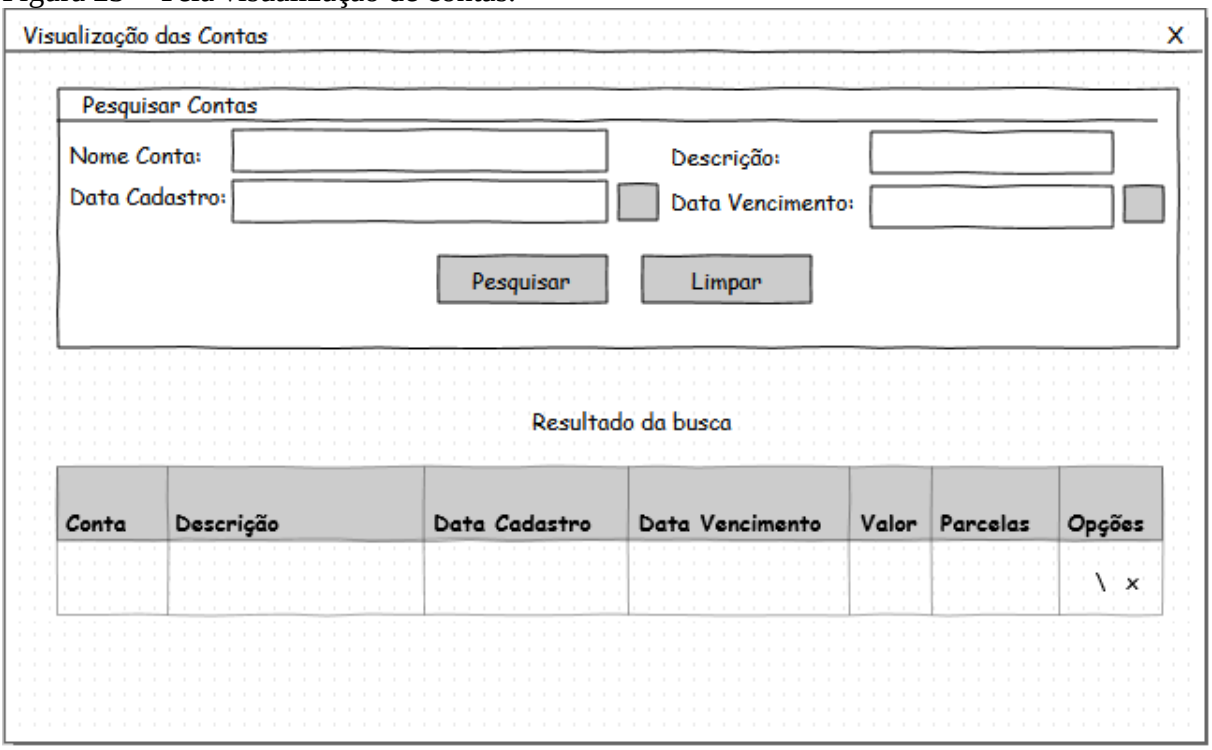
Figura 22 – Tela cadastro e edição.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 23 vem ilustrar a tela de visualização de contas, onde o usuário terá a opção de efetuar as buscas utilizando dos filtros para refinar melhor a sua pesquisa. Após a realização da busca, a conta poderá ser editada ou excluída.

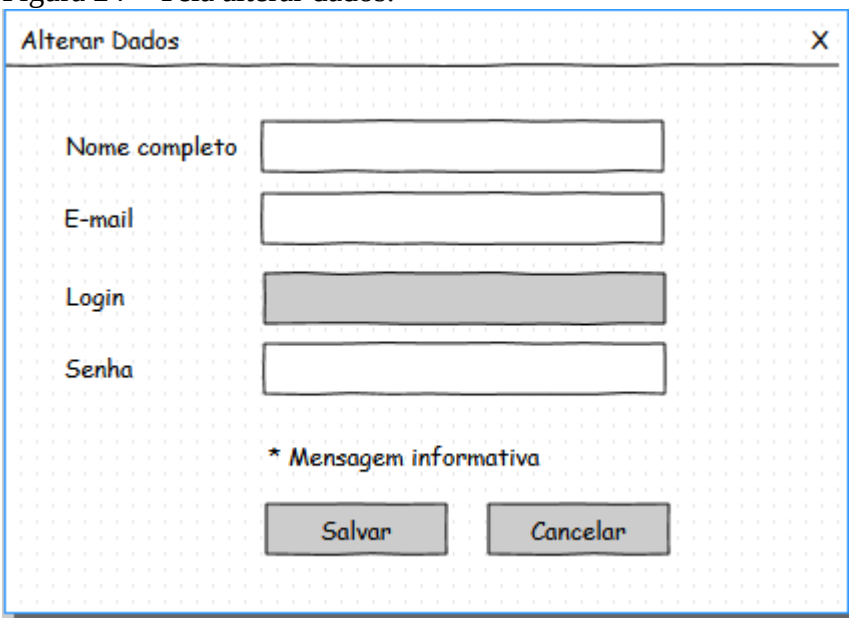
Figura 23 – Tela visualização de contas.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 24 contempla a tela de alteração de dados, nessa tela o usuário poderá alterar seus dados realizados no cadastro, o campo *login* não poderá ser editado e ao final da ação será informada uma mensagem de sucesso ou erro no campo mensagem informativa.

Figura 24 – Tela alterar dados.



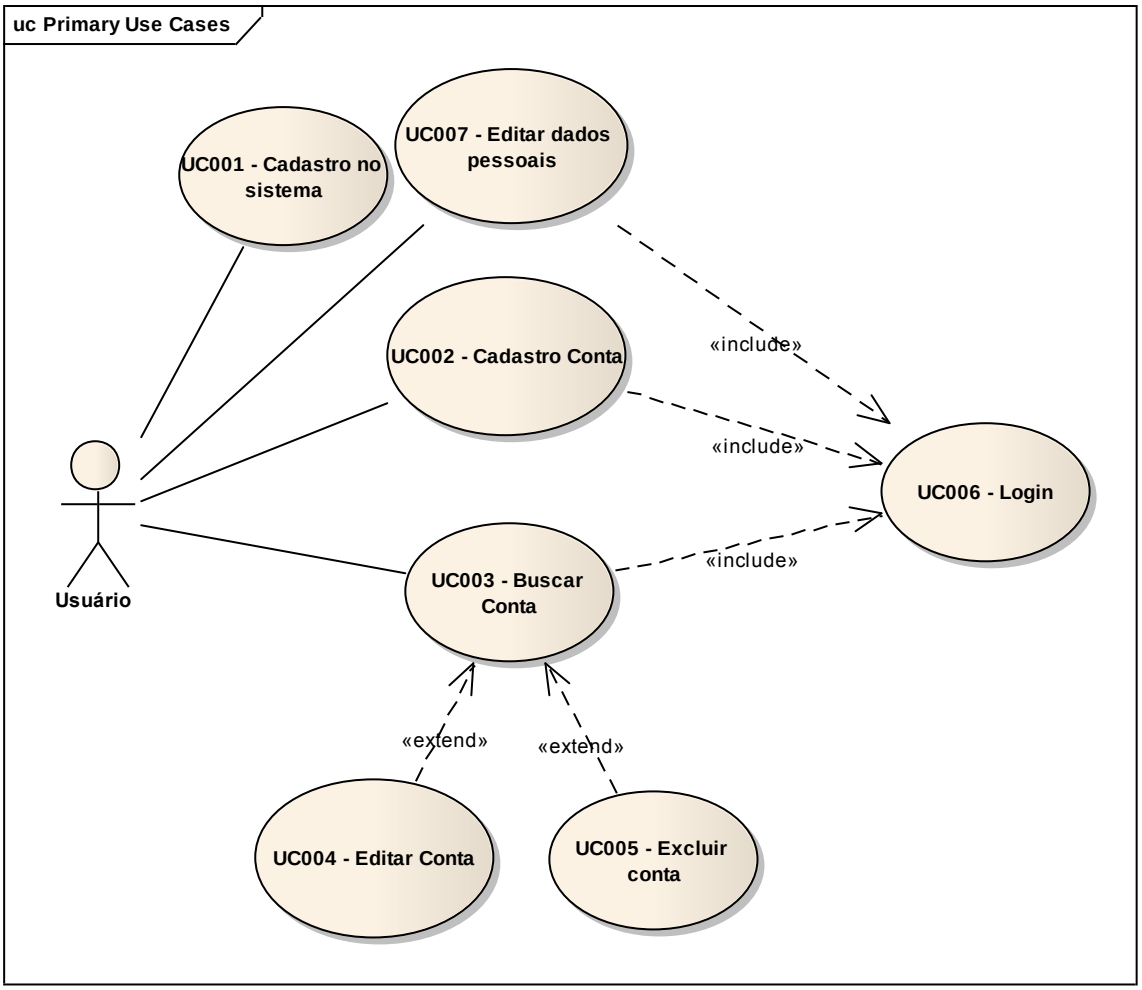
Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

4.2.4 CASOS DE USO PRIMÁRIO

De acordo com Maia (2005), o caso de uso é usado para representar de forma clara e legível todos os cenários que o usuário irá executar em alguma de suas tarefas.

A figura 25 ilustra a interação dos casos de usos

Figura 25 – Caso de uso.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A Tabela 2 mostra o significado de cada caso de uso.

Quadro 2– Descrição dos Casos de Uso

Número Caso de Uso	Descrição
UC001 – Cadastro no sistema	Opção para cadastrar novos usuários para o sistema
UC002 – Cadastro Conta	Opção para cadastro das contas no sistema
UC003 – Excluir conta	Opção para excluir contas já cadastra no sistema
UC004 – Buscar Conta	Opção para visualizar todas as contas do usuário cadastradas no sistema
UC005 – Editar Conta	Opção onde o usuário poderá editar as informações das contas
UC006 – Login	O usuário deve informar o login e senha para acessar do sistema
UC007 – Editar dados Pessoais	Opção para editar os dados pessoais do usuário e a senha de acesso no sistema.

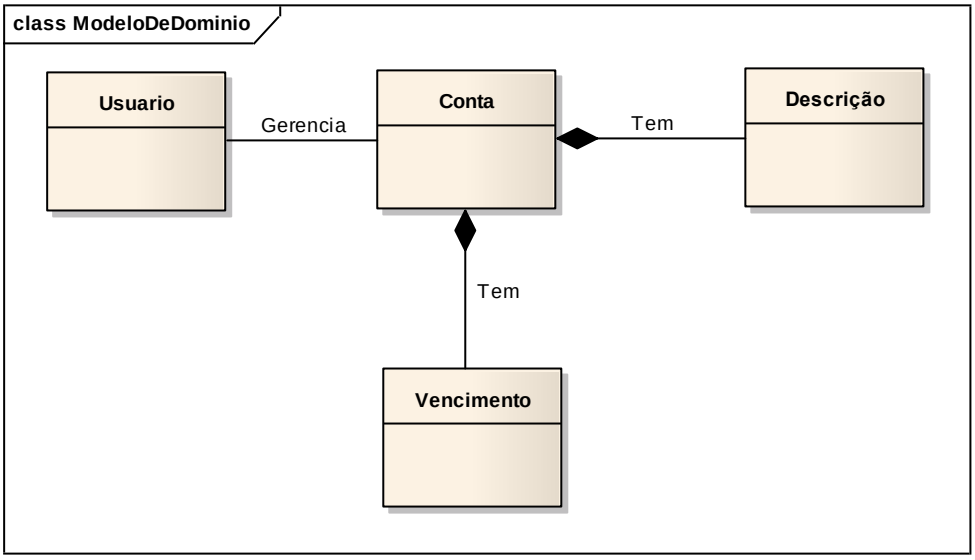
Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

4.2.5 MODELO DE DOMÍNIO

Sendo Silva, G et. al. (2007) O modelo de domínio é construído a partir dos requisitos do sistema, de modo a achar as classes, substantivos, frases de substantivos, verbos e frases de verbos, que se tornarão objetos, atributos e associações em um diagrama.

A figura 26 apresenta o modelo de domínio do sistema proposto, de forma que o usuário pode possuir uma ou varias contas e as contas são gerenciadas por um usuário.

Figura 26 – Modelo de Domínio.



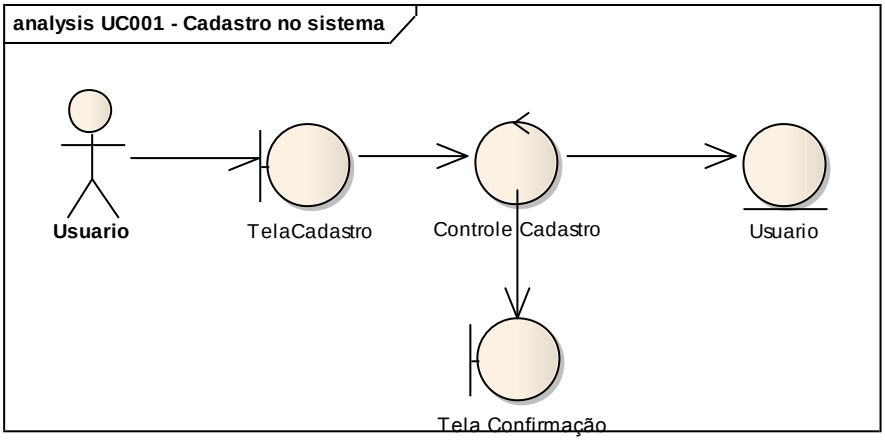
Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

4.2.6 DIAGRAMA DE ROBUSTEZ

Conforme Silva, G et. al. (2007) "O diagrama de robustez tem por finalidade conectar a parte de análise com a parte de projeto, assegurando que a descrição dos casos de uso esteja correta e, dentro da ideia proposta, realizando o mapeamento dos objetos do diagrama".

A figura 27 retrata o diagrama de robustez referente ao caso de uso UC001 – Cadastra no Sistema.

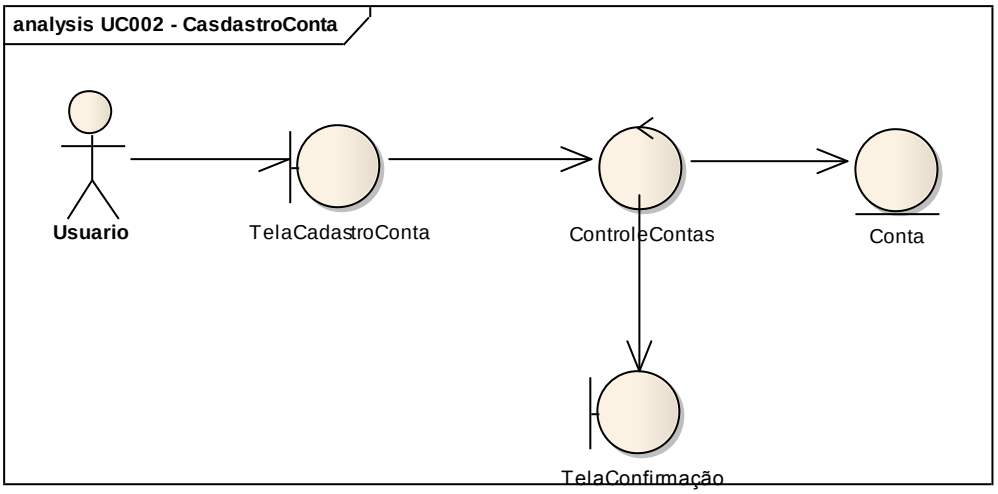
Figura 27 – Diagrama de Robustez do Cadastro no Sistema



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 28 mostra o diagrama de robustez referente ao caso de uso UC002 – Cadastra Conta.

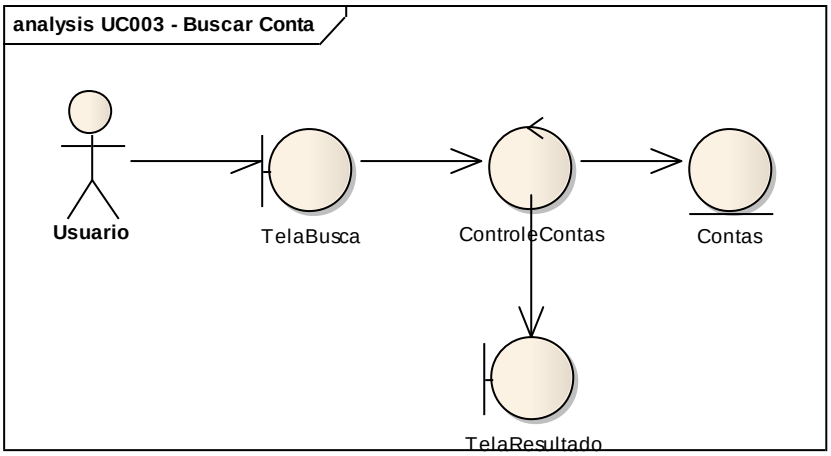
Figura 28 – Diagrama de Robustez do Cadastro Conta



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 29 retrata o diagrama de robustez referente ao caso de uso UC003 – Buscar Conta.

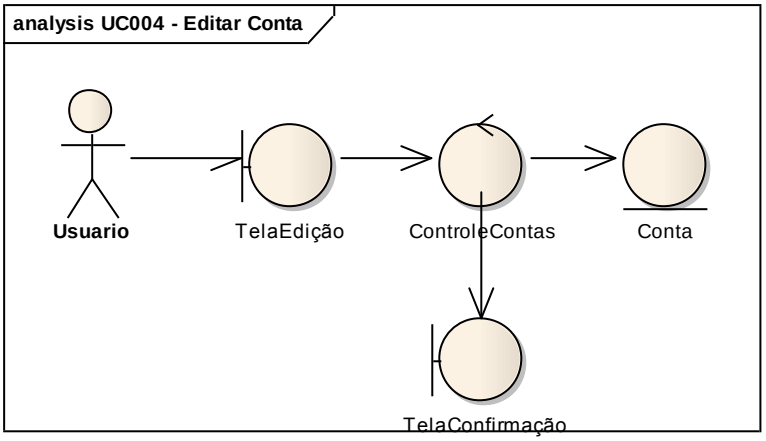
Figura 29 – Diagrama de Robustez para Buscar Conta



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 30 apresenta o diagrama de robustez referente ao caso de uso UC004 – Editar Conta.

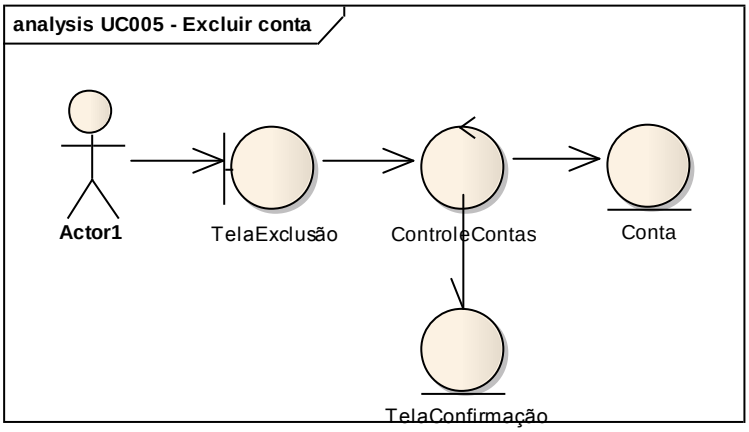
Figura 30 – Diagrama de Robustez para Editar Conta



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 31 mostra o diagrama de robustez referente ao caso de uso UC005 – Excluir Conta.

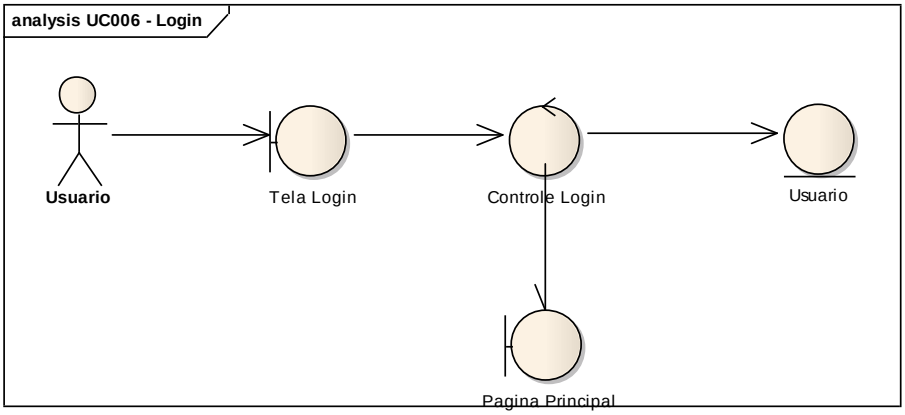
Figura 31 – Diagrama de Robustez para Excluir Conta



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 32 retrata o diagrama de robustez referente ao caso de uso UC006 – *Login*.

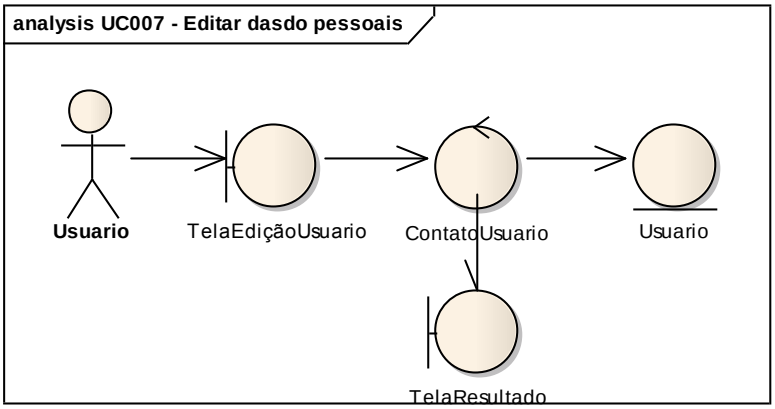
Figura 32 – Diagrama de Robustez Login



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 33 apresenta o diagrama de robustez referente ao caso de uso UC007 – Editar Dados Pessoais.

Figura 33 – Diagrama de Robustez para Editar Dados Pessoais



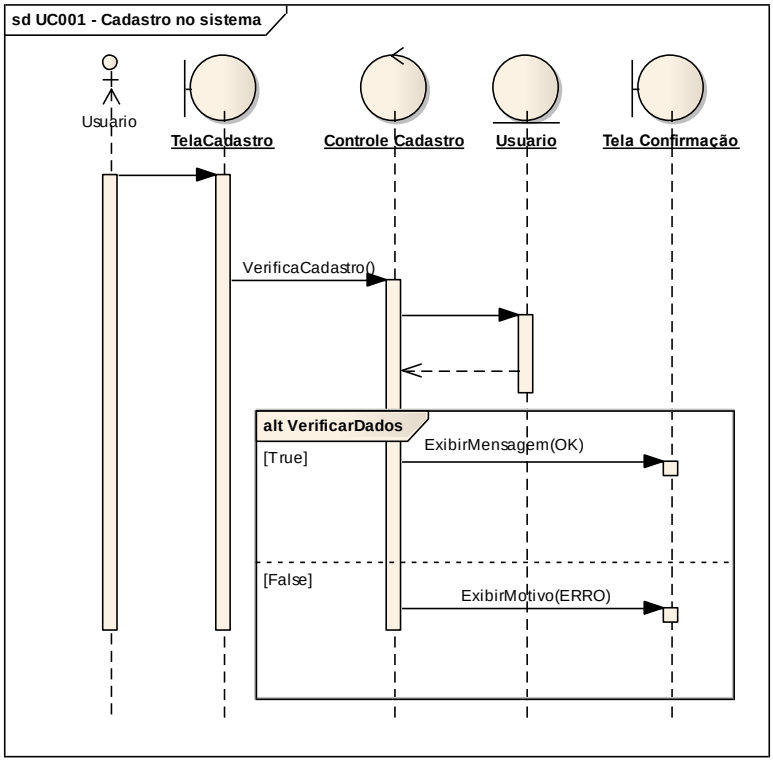
Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

4.2.7 DIAGRAMA DE SEQUÊNCIA

O diagrama de sequência consiste em construir um modelo entre o usuário e o sistema, tendo como papel principal mostrar o funcionamento do sistema em tempo de execução (MAIA, 2005).

A figura 34 retrata o diagrama de sequência referente ao caso de uso UC001 – Cadastro no sistema.

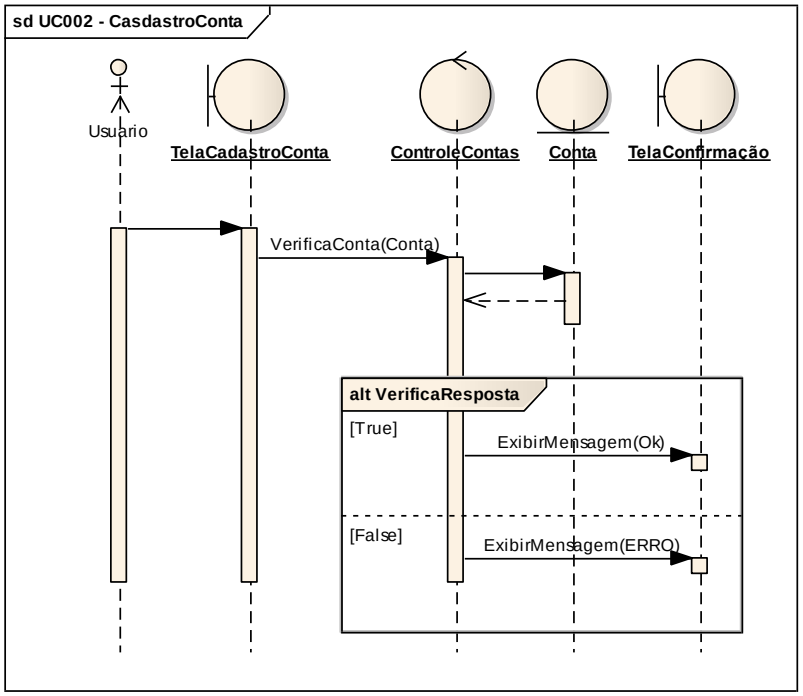
Figura 34 – Diagrama de Sequência do Cadastro no Sistema



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 35 apresenta o diagrama de sequência referente ao caso de uso UC002 – Cadastro Conta.

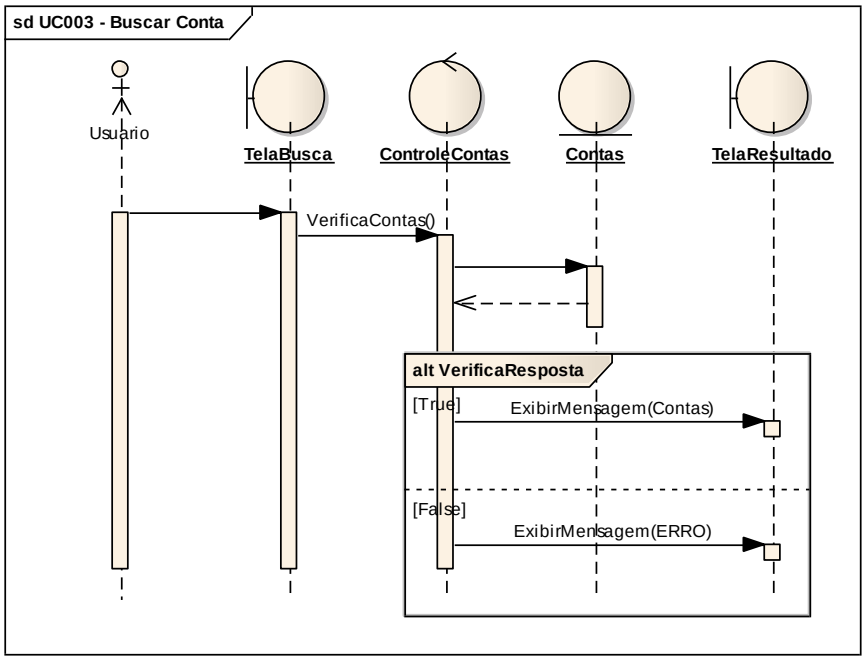
Figura 35 – Diagrama de Sequência do Cadastro de Contas



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 36 expõe o diagrama de sequência referente ao caso de uso UC002 – Busca Conta.

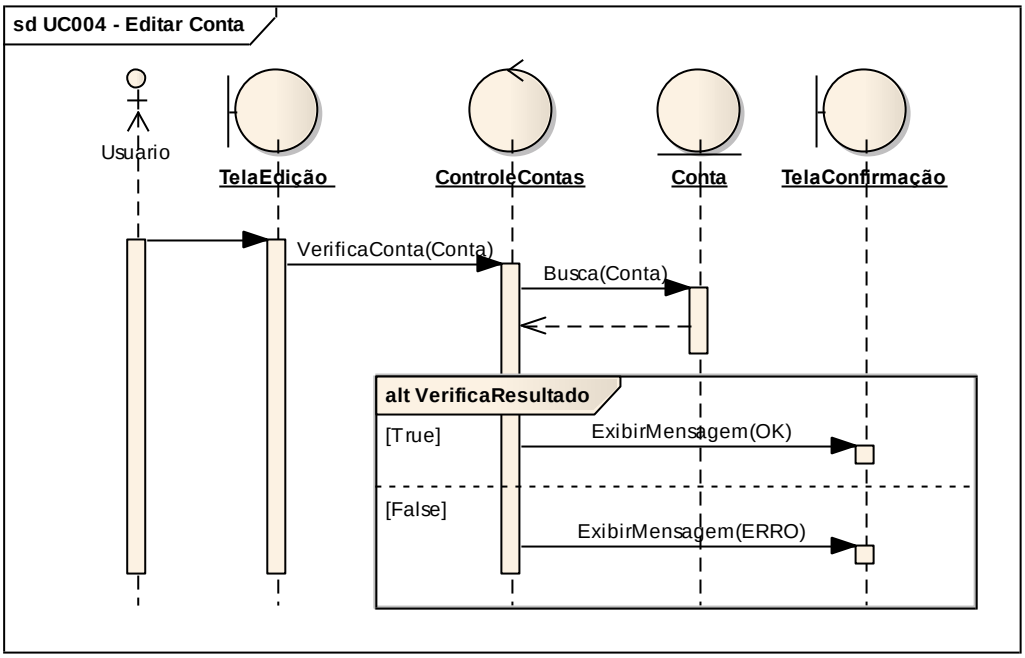
Figura 36 – Diagrama de Sequência da Busca de Contas



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 37 exibe o diagrama de sequência referente ao caso de uso UC004 – Editar Conta.

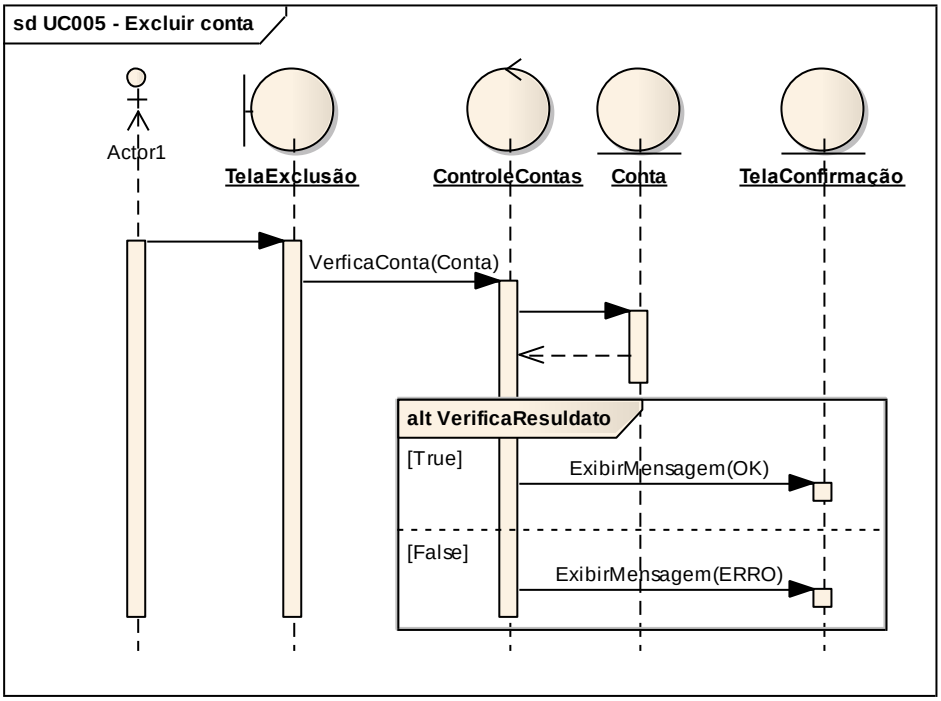
Figura 37 – Diagrama de Sequência Cadastro no Sistema



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 38 mostra o diagrama de sequência referente ao caso de uso UC005 – Excluir Conta.

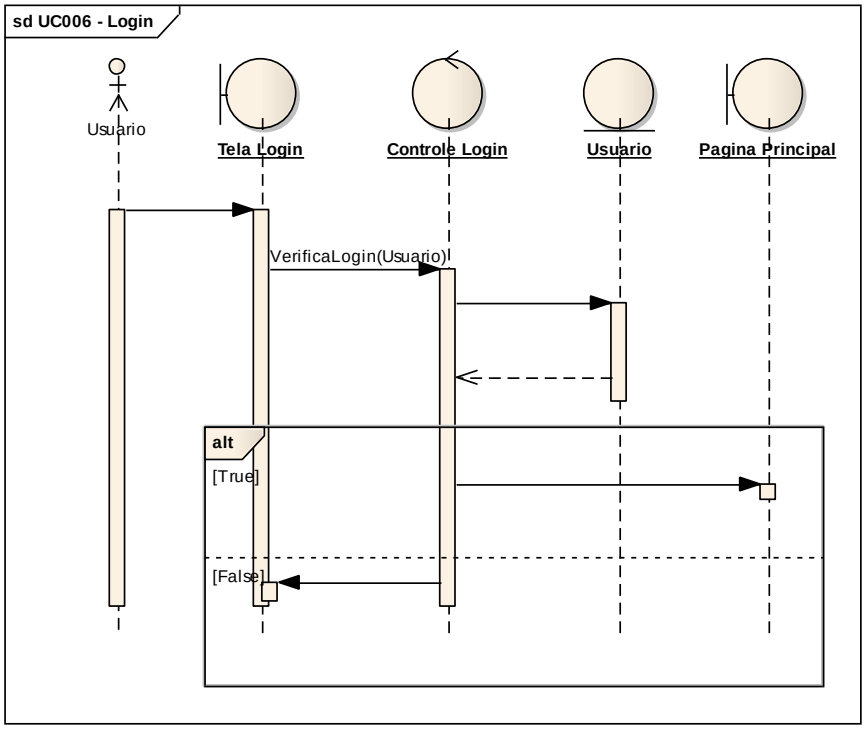
Figura 38 – Diagrama de Sequência para Excluir Contas



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 39 apresenta o diagrama de sequência referente ao caso de uso UC006 – Login.

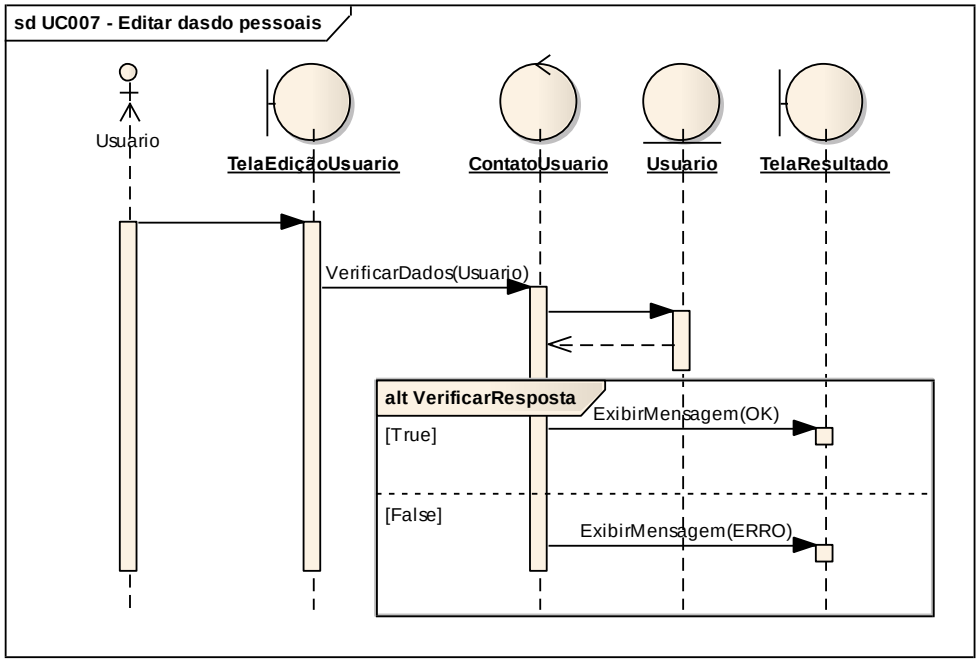
Figura 39 – Diagrama de Sequência de Login



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 40 contempla o diagrama de sequência referente ao caso de uso UC007 – Editar dados pessoais.

Figura 40 – Diagrama de Sequência para Editar Dados Pessoais



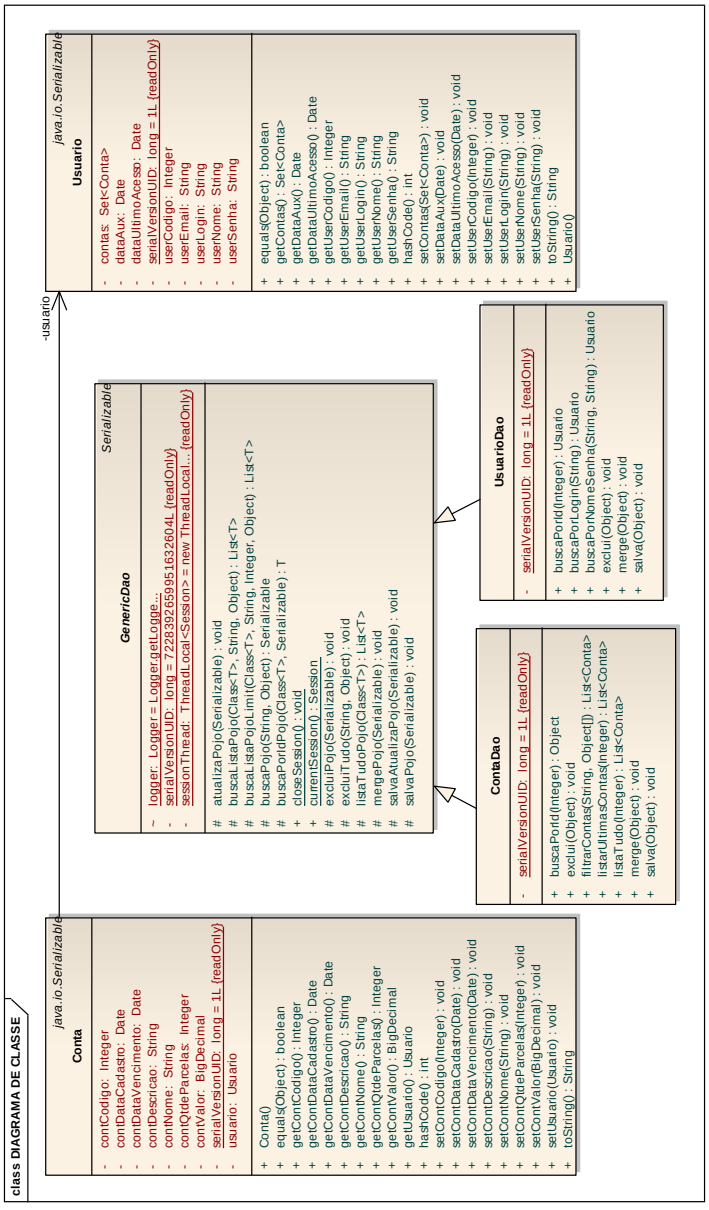
Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

4.2.8 DIAGRAMA DE CLASSE

O diagrama de classe descreve os tipos de objetos presentes no sistema e seus vários tipos de relacionamentos estáticos existentes. O diagrama de classe também mostra a propriedade e as operações de uma classe e as restrições que se aplicam à maneira como os objetos estão conectados. (COSTA, 2007).

A figura 41 expõe o diagrama de classe do sistema proposto.

Figura 41 – Diagrama de Classe



Fonte: (ELABORAÇÃO DOS AUTORES, 2012)

4.3 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Este capítulo, abordou o desenvolvimento da modelagem do projeto de solução, sendo utilizado para o desenvolvimento o ICONIX, onde reúne um conjunto de processos que mostram as visões dinâmicas e estáticas do sistema.

5 SISTEMA DESENVOLVIDO

Neste capítulo, será apresentado o desenvolvimento do sistema proposto no capítulo quatro e as ferramentas necessárias para deixar rodando em ambiente normal, tendo em vista que o foco do trabalho não é no desenvolvimento do sistema, mas sim na comparação entre os ambientes disponibilizados pela Amazon (IaaS) e pela Google (PaaS).

5.1 ESTUDO DE CASO

Esta seção aborda os seguintes módulos: cenário proposto, ferramentas utilizadas, esquema do sistema, protótipo do cenário, validação e resultados.

5.1.1 Cenário proposto

O cenário escolhido para o estudo de caso proposto foi o desenvolvimento de um sistema para controle de contas, tendo por finalidade facilitar o gerenciamento de suas contas pessoais, de modo que o usuário poderá cadastrar, visualizar, editar ou excluir contas.

O cenário proposto primeiramente foi desenvolvido em um ambiente tradicional, a partir disto foi migrado para a nuvem da Amazon[1] (IaaS) e o ambiente da Google[2] (PaaS).

[1] *http://aws.amazon.com/pt/*

[2] *https://appengine.google.com/*

5.1.1.1 Ferramentas utilizadas

Nesta seção são apresentadas as ferramentas utilizadas para o desenvolvimento do sistema proposto em um ambiente local.

5.1.1.1.1 Java

O Java é uma linguagem de programação criada pela Sun Microsystems em 1991 a partir do C++, tendo como objetivo criar uma nova geração de computadores portáteis inteligentes, capazes de se comunicar de muitas formas, ampliando suas potencialidades de uso. Com estes motivos, decidiu-se criar uma nova plataforma para o desenvolvimento destes equipamentos de forma que seu software pudesse ser portado para os mais diferentes tipos de equipamentos. (JANDL, 1990).

A linguagem Java tem as principais características que se diferencia das outras linguagens de programação, abaixo algumas das características (JANDL, 1990):

- Orientação a objetos;
- Independência de Plataformas;
- Sem Ponteiros;
- Desempenho
- Segurança
- Permite Multithreading

Mas para que se possa executar uma aplicação Java é necessário se te uma Máquina Virtual do Java (JVM), pois o JVM se comunica com os diferentes tipos de sistemas operacionais fazendo com que a linguagem se comunique diretamente com ela, não havendo a necessidade de cria um arquivo diferente para cara sistema operacional. (SALVADOR, 2008).

5.1.1.1.2 JSF2

Conforme Pitanga (2011) o JavaServer Faces (JSF) é

> Uma tecnologia que incorpora características de um framework MVC para WEB e de um modelo de interfaces gráficas baseado em eventos. Por basear-se no padrão de projeto MVC, uma de suas melhores vantagens é a clara separação entre a visualização e regras de negócio.

O MVC isola a lógica de negócio da lógica de apresentação de uma aplicação, dividida em três camadas (K19, 2012):

- Modelo: encapsula os dados e as funcionalidades da aplicação
- Visão: é responsável pela exibição de informações, cujos dados são obtidos do modelo.
- Controlador: recebe as requisições do usuário e aciona o modelo e/ou a visão

5.1.1.1.3 JPA

Segundo Paes (2007) o JPA é

> Um framework utilizado na camada de persistência para o desenvolvedor ter uma maior produtividade, com impacto principal num modo para controlarmos a persistência dentro de Java. Pela primeira vez, nós, desenvolvedores temos um modo "padrão" para mapear nossos objetos para os do Banco de Dados. Persistência é uma abstração de alto-nível sobre JDBC.

5.1.1.1.4 Hibernate

De acordo com Rocha et al (2011) o hibernate é:

> Um tipo de framework para o mapeamento objeto-relacional totalmente desenvolvido na linguagem Java. Este programa visa facilitar o mapeamento dos atributos entre uma base de dados tradicional e o modelo orientado a objetos de uma aplicação, mediante o uso de arquivos (XML) para estabelecer esta relação.

5.1.1.1.5 TomCat 7

Segundo a Apache Software Foundation(2012), o TomCat é uma implementação aberta do Java Servlet e JavaServer Pages, ambas desenvolvidos sob as especificações do Java Community Process. O TomCat é um servidor web desenvolvido em Java que permite a execução de aplicativos desenvolvidos em Java nas tecnologias de Servlets e Java Server Pages.

5.1.1.1.6 MYSQL

O MYSQL foi criado na década de 90, e é considerado como um servidor e um gerenciador de banco de dados relacional, sendo criado para trabalho com aplicações de pequeno a médio porte, algo em torno de 100 milhões de registros por tabela, tendo como tamanho médio aproximadamente 100MB por tabela. (MILANI, 2012).

Também possui características de um SGBD (Sistema Gerenciador de Banco de Dados) que "consiste no software que gere todo o acesso a uma ou mais bases de dados, permitindo a definição, acesso concorrente, manipulação e controlo dos dados, assegurando a integridade, segurança e recuperação das bases de dados". (NEVES et al, 2005)

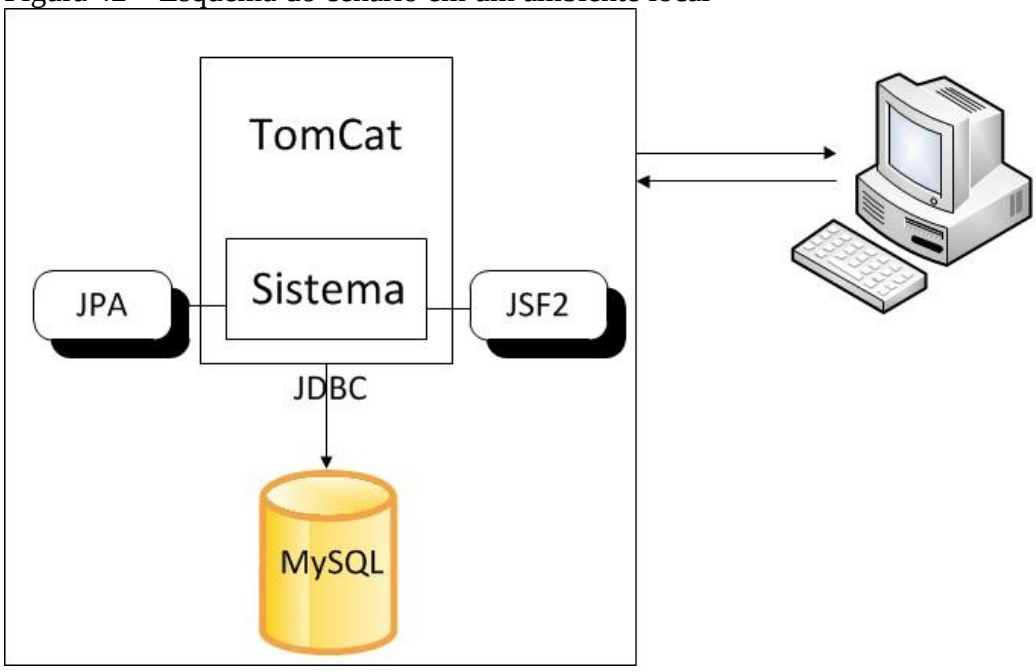
5.1.1.1.7 Eclipse

O eclipse consiste em um Ambiente Integrado de Desenvolvimento (*Integrated Development Environment* - IDE), podendo ser utilizado para desenvolver software em qualquer linguagem. Sendo criado pela International Business Machine (IBM) com o intuito de substituir o seu produto Visual Age for Java, mas em novembro de 2001 teve o seu código fonte liberado e se tornado uma aplicação *open source.* (BURNETTE, 2006).

5.1.1.2 Esquema do sistema

Para deixar o sistema rodando em um ambiente web tradicional é necessário ter a estrutura conforme a figura 42.

Figura 42 – Esquema do cenário em um ambiente local



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 42 mostra a infraestrutura necessária para se ter um sistema executando em um ambiente web tradicional, sendo necessário um servidor com o *TomCat*, *MySQL* e o *Java* instalados para a execução do sistema proposto.

5.1.1.3 Protótipo do cenário

Esta seção ira apresentar as telas desenvolvidas através dos protótipos de telas que foram projetadas no capítulo quatro.

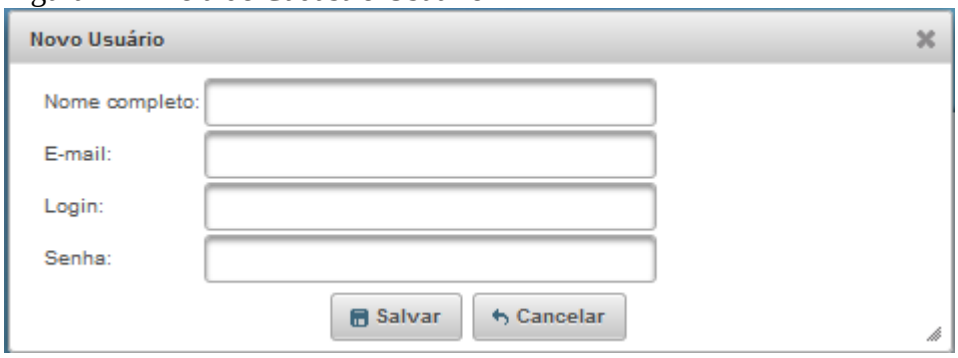
A figura 43 acima apresenta a tela inicial do sistema.

Figura 43 – Tela de Login

Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 44 acima corresponde a tela de cadastro de um novo usuário.

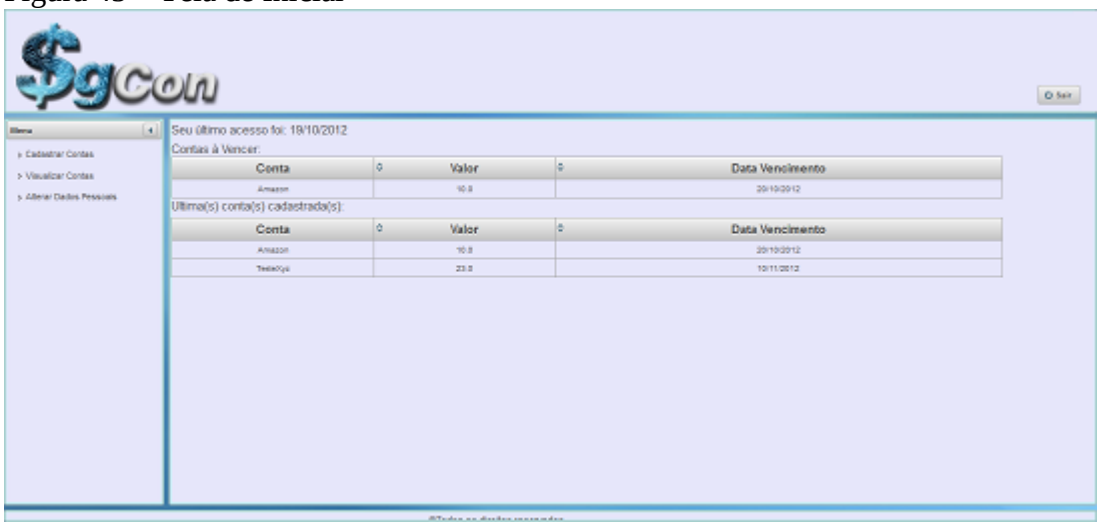
Figura 44 – Tela de Cadastro Usuário



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 45 ilustra a tela inicial do sistema, nela estão contidas informações sobre o último acesso, as contas próximas a vencerem e as últimas contas cadastradas.

Figura 45 – Tela de Inicial



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

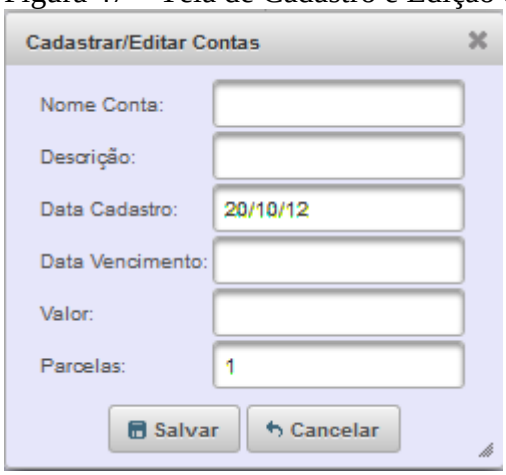
A figura 46 contempla a tela para a pesquisa dentre todas as contas cadastradas, podendo também realizar alguns filtros para a exibição das contas. Nesta tela é possível selecionar uma conta para a edição ou excluí-la conforme pode ser observado na coluna opções.

Figura 46 – Tela de Visualização de Contas

Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 47 ilustra a tela de Cadastro/Edição de contas, quando o usuário acionar o botão de edição na tela de visualização de contas representada pela figura 45 a tela virá com os dados já preenchidos, quando o usuário escolher a opção *Cadastrar Contas* no menu lateral esquerdo representado na figura 44 a tela virá apenas com a data de cadastro com o dia atual e quantidade de parcelas com o número 1, necessitando a conclusão do preenchimento dos outros campos.

Figura 47 – Tela de Cadastro e Edição de contas



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 48 contempla a tela de alteração de dados, que conforme as regras não é possível alterar apenas o campo de *Login*.

Figura 48 – Tela de Edição Dados do Usuário

Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

5.1.2 Amazon Web Services

A *Amazon Web Services* oferece um conjunto completo de serviços de aplicativos e infraestrutura que permitem que seja executado quase tudo na nuvem, desde aplicativos empresariais e projetos de dados grandes a jogos sociais e aplicativos móveis.

Segundo a Amazon(2012) "...alguns dos benefícios essenciais da computação em nuvem é a opção de substituir os gastos com a infraestrutura principal por preços variáveis baixos, que se adaptam de acordo com a empresa". A *Amazon* dispõe de uma variedade de serviços baseados na nuvem disponíveis através do portal da *Amazon Web Services* (AWS). Dentre os serviços computacionais ou de rede ofertados pela *Amazon* foi utilizado, para esse estudo de caso, o *Elastic Compute Cloud (EC2)*, o qual será detalhado no tópico a seguir.

5.1.2.1.1 Amazon Elastic Compute Cloud (EC2)

O serviço EC2 da Amazon permite que você crie máquinas virtuais (instâncias) e as gerencie de acordo com a necessidade, podendo configurar e instalar o que for necessário, onde o administrador da instância tem o total acesso a raiz de cada máquina e assim podendo interagir com ela como se fosse uma máquina local. Este serviço também possui uma alta flexibilidade, onde o administrador pode optar dentre várias instâncias, sistemas operacionais e pacotes de software. O Amazon EC2 permite que você selecione uma configuração de memória, CPU, armazenamento de instância e tamanho da partição de inicialização que seja ideal para a sua opção de sistema operacional e aplicativos.

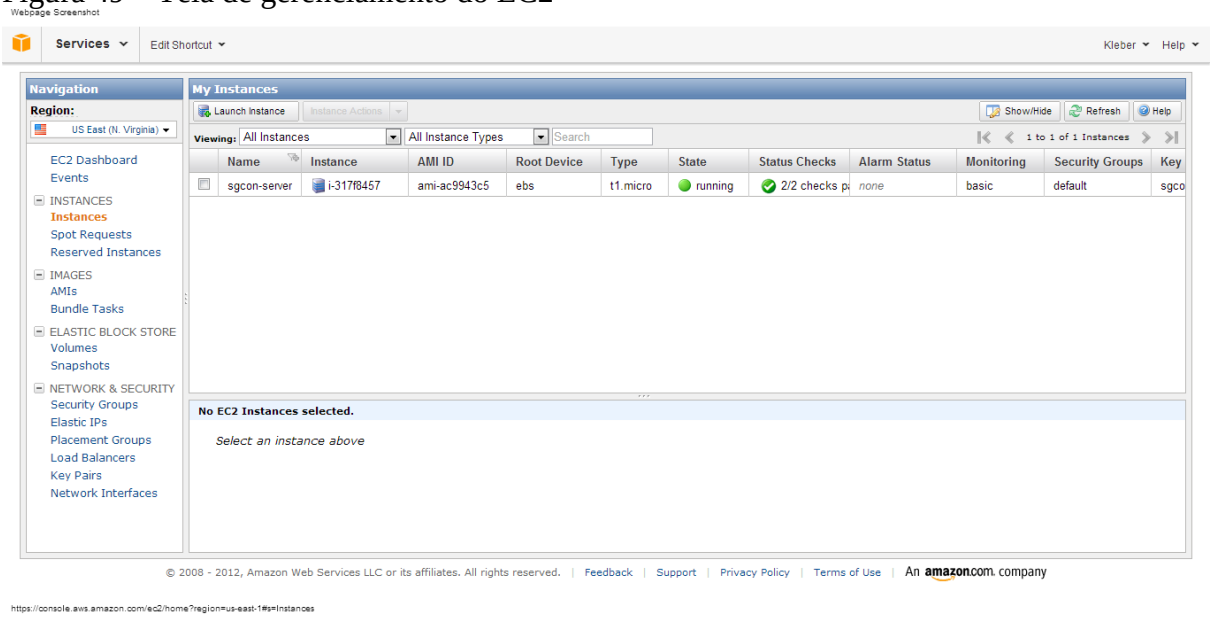
Segundo a Amazon (2012) o EC2 é um serviço da Web que fornece uma capacidade de computação redimensionável na nuvem, e foi projetada para tornar a escalabilidade computacional de nível de web mais fácil para desenvolvedores.

O EC2 é gerenciado no portal AWS através de um painel de controle chamado de *AWS Management Console*, uma interface simples que permite que seja feito o controle completo de todos os recursos computacionais disponibilizados da Amazon. Este painel de controle reduz o tempo exigido para criar e inicializar as novas instâncias do servidor,

permitindo que seja escalonado a capacidade, para mais ou para menos, à medida que os requisitos forem alterados. Para a Amazon (2012) o EC2 prove uma economia ao permitir que seja pago somente pela capacidade que realmente utilizar.

A figura 49 abaixo apresenta o painel de controle responsável pelo gerenciamento dos recursos e instâncias.

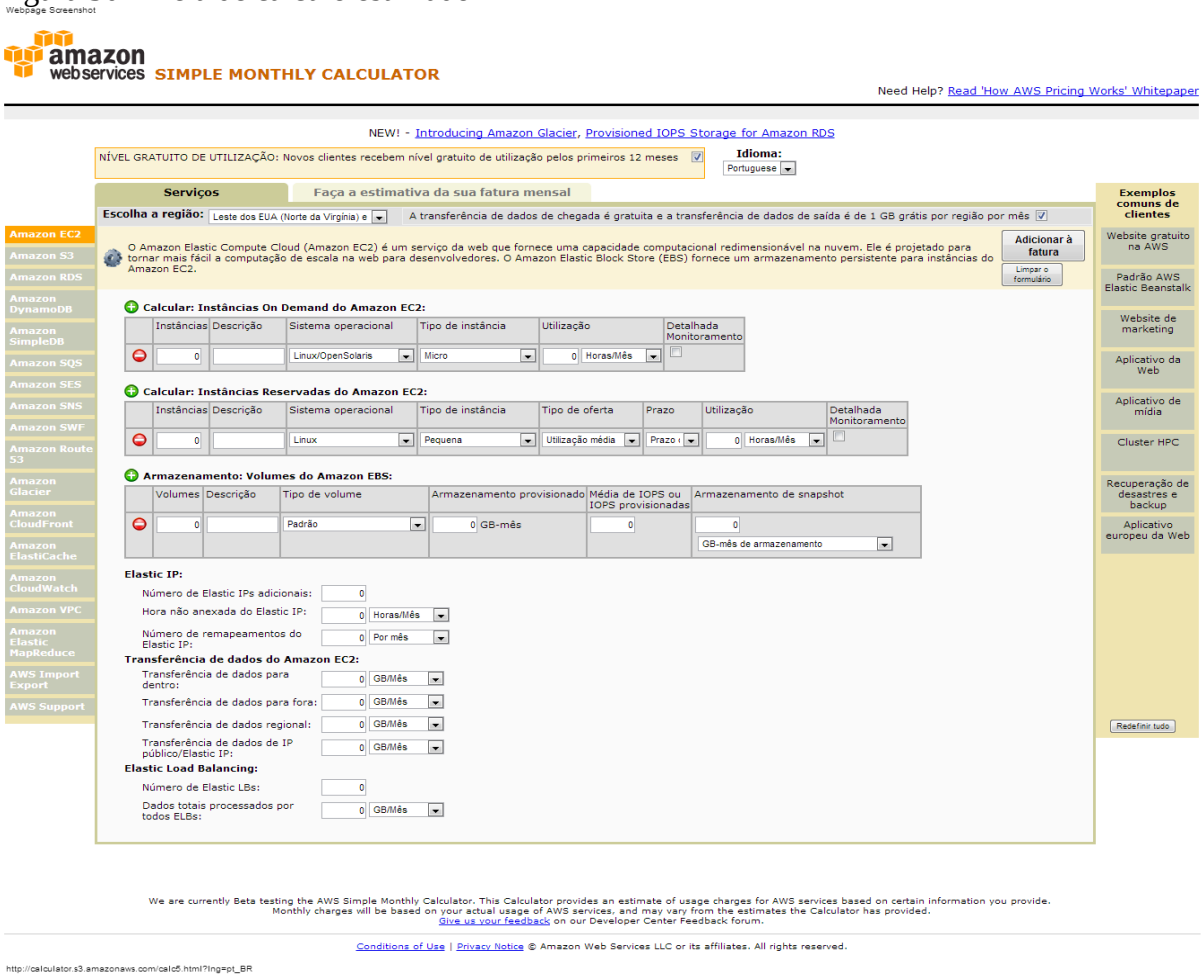
Figura 49 – Tela de gerenciamento do EC2



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A Amazon disponibiliza um serviço chamado “Calculador Mensal Simples” onde pode ser feito um cálculo estimado de quanto custará mensalmente de acordo com o tipo de instância, memória, CPU, e utilização/hora mensal. A figura 50 a seguir ilustra este serviço.

Figura 50 – Tela de cálculo estimado

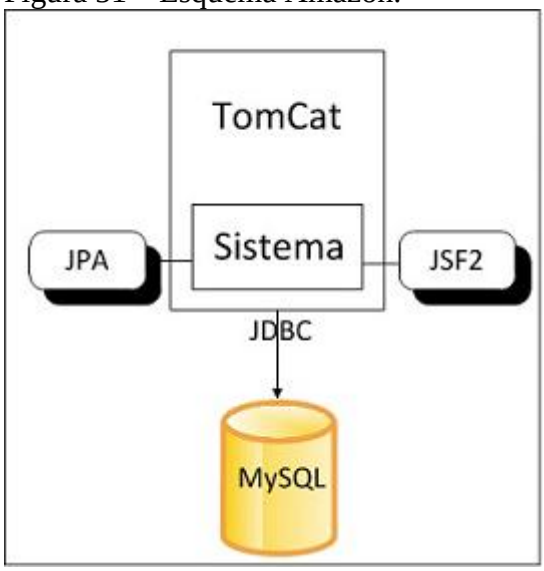


Fonte: (http://calculator.s3.amazonaws.com/calc5.html?lng=pt_BR, acesso em: 2012).

5.1.2.2 Esquema da Solução

Para o sistema proposto o esquema da solução ficou montado conforme ilustrado pela figura 51 a seguir.

Figura 51 – Esquema Amazon.

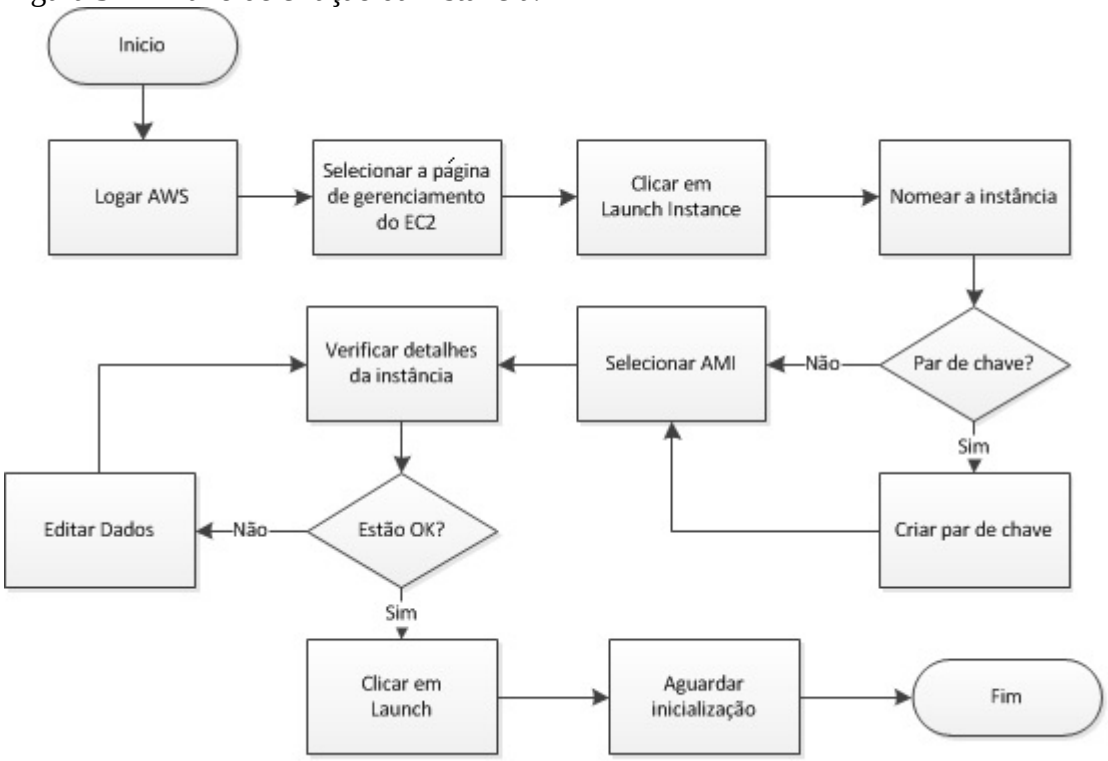


Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

5.1.2.3 Preparação do ambiente

Para a criação de uma instância e necessário seguir um fluxo para que essa máquina virtual seja criada e iniciada. A figura 52 abaixo ilustra esse fluxo, e a seguir será explanado cada passo com figuras demonstrativas sobre cada um.

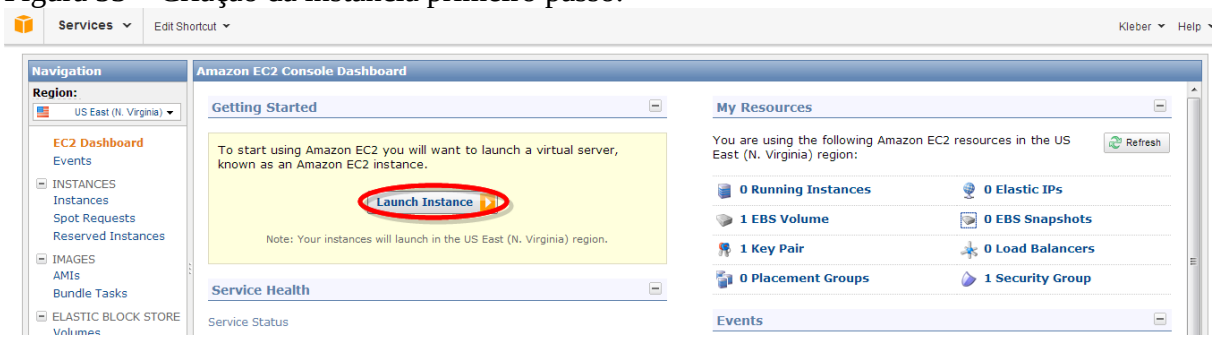
Figura 52 – Fluxo de criação da instância.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 53 abaixo ilustra o primeiro passo para a criação de uma instância.

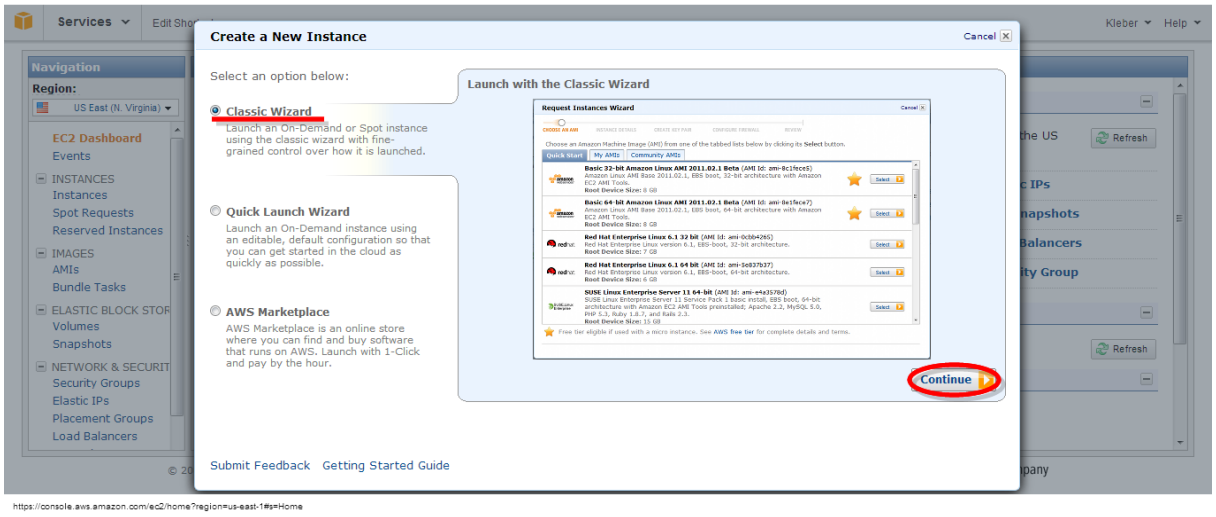
Figura 53 – Criação da instância primeiro passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

O segundo passo é escolher o tipo de criação de uma instância, que neste caso foi o *Classic Wizard* seguido pelo botão *Continue*, conforme a figura 54 a seguir.

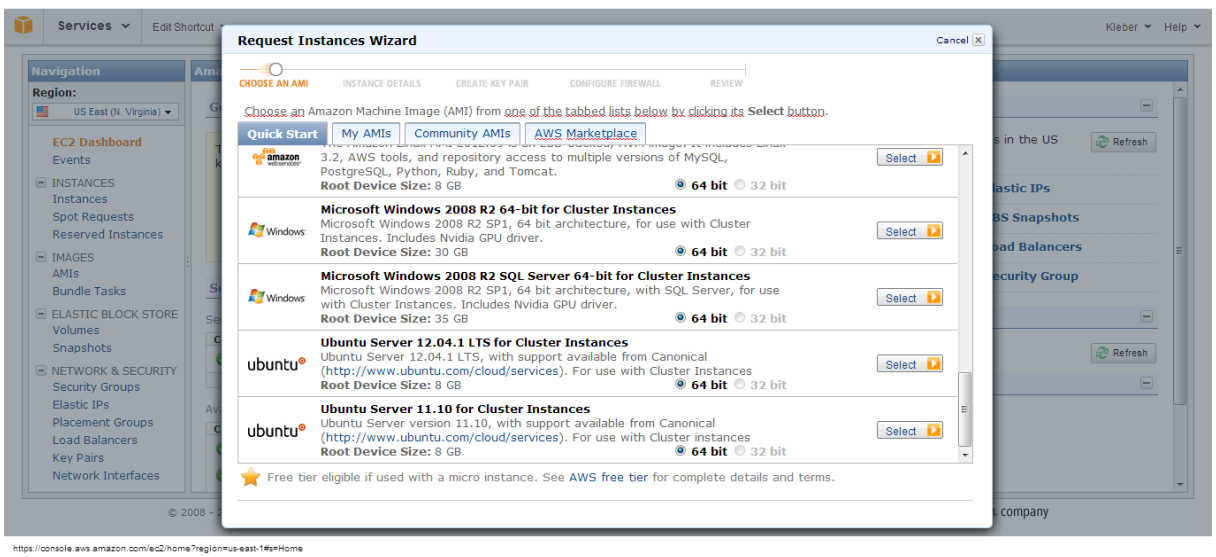
Figura 54 – Criação da instância segundo passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

No terceiro passo é necessário definir o tipo de sistema operacional para a instância, há neste passo dois tipos de sistemas para escolher, um sistema operacional comum comumente usado em uma máquina local ou um sistema operacional próprio para o ambiente de *cluster*, onde esses sistemas são otimizados para o funcionamento nesta modalidade. Há também a opção de usar imagens criadas anteriormente pelo usuário na aba "*My AMIS*" ou então seguir para a aba "*Community AMIS*" e usar imagens criadas por outros usuários com alguns pacotes de softwares, e configurações previamente instaladas. No entanto para cada tipo de sistema operacional é definido um valor, em dólares, por hora de funcionamento da instância. A figura 55 a seguir ilustra esta tela.

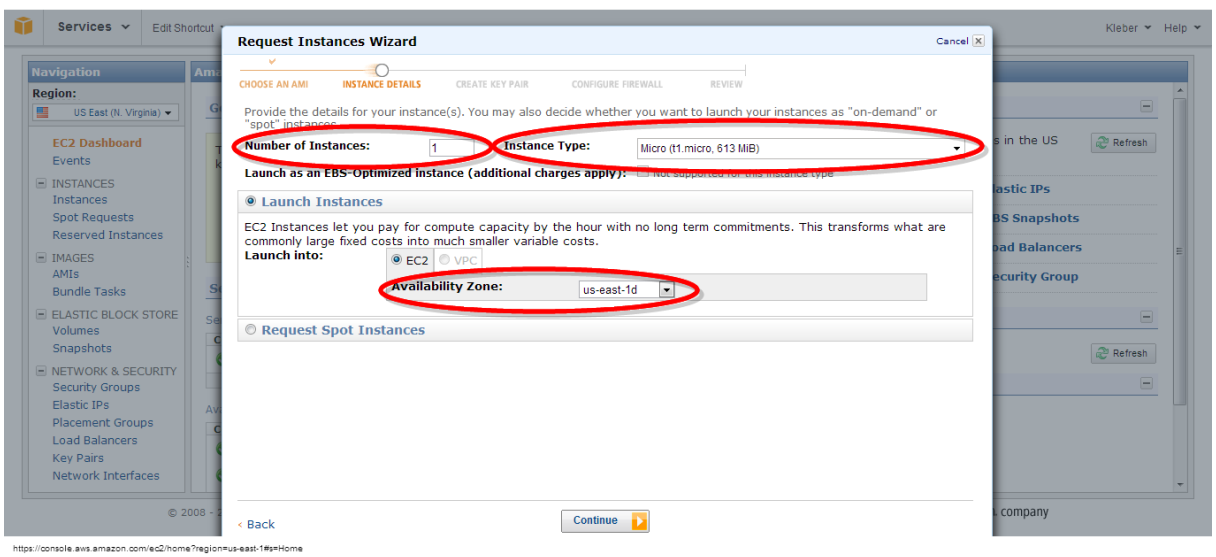
Figura 55 – Criação da instância terceiro passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

Para o quarto passo é necessário definir quantas instâncias serão criadas, o tipo de instância, e o local físico de onde ficará a máquina virtual. É importante salientar que o custo para entrada e saídas de dados da instância é de acordo com o local físico definido na criação neste passo. A figura 56 abaixo ilustra esta tela.

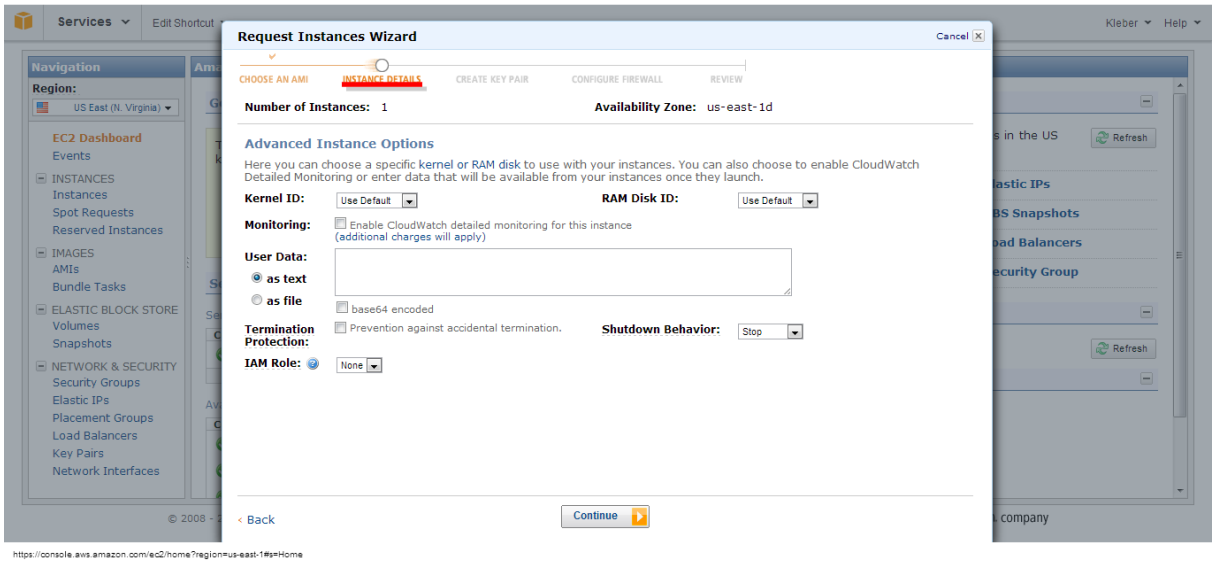
Figura 56 – Criação da instância quarto passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A figura 57 a seguir ilustra a tela com alguns detalhes, e algumas opções avançadas da instância no quinto passo.

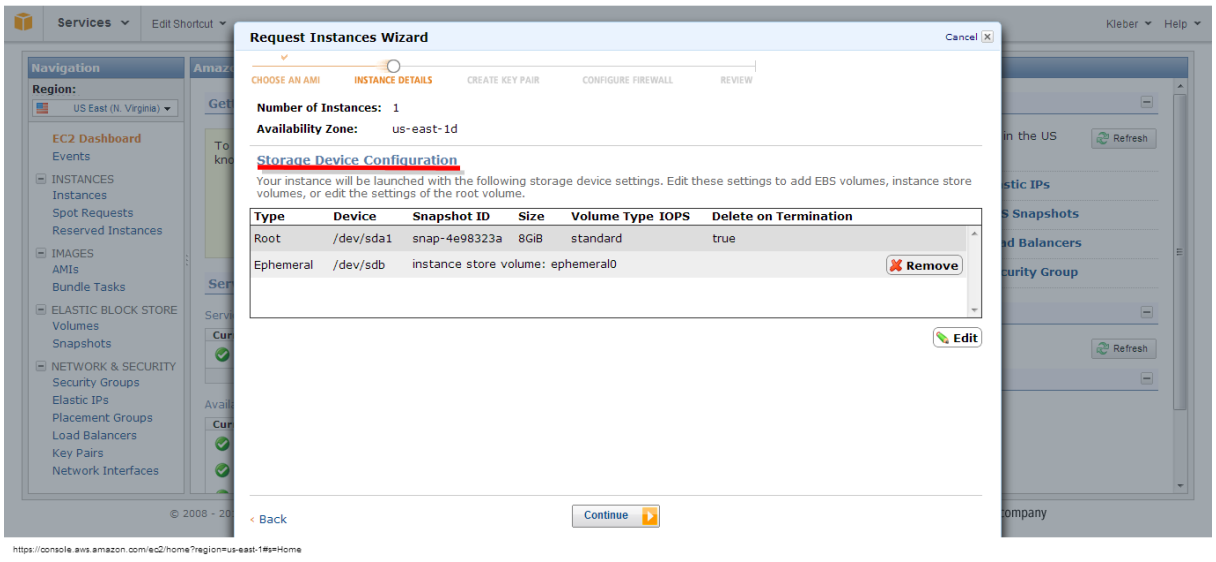
Figura 57 – Criação da instância quinto passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A próxima figura 58 ilustra o sexto passo com as configurações do tipo de armazenamento para a instância.

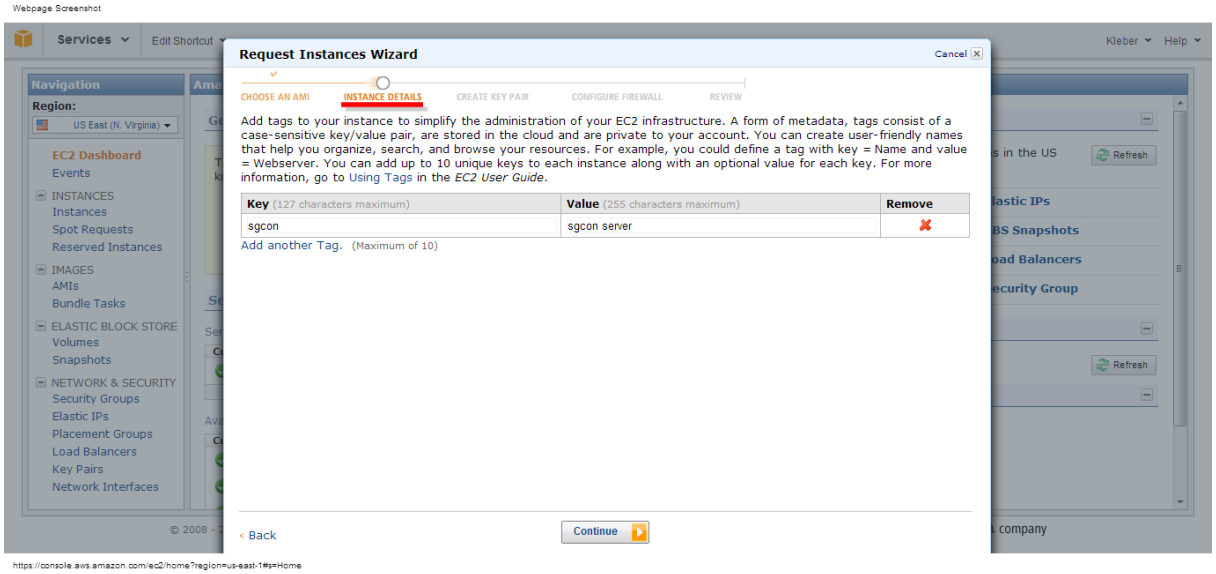
Figura 58 – Criação da instância sexto passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

No sétimo passo tem a opção de adicionar "*tags*", ou seja, identificadores para simplificar a administração das instâncias criadas. A figura 59 abaixo a contempla esta tela.

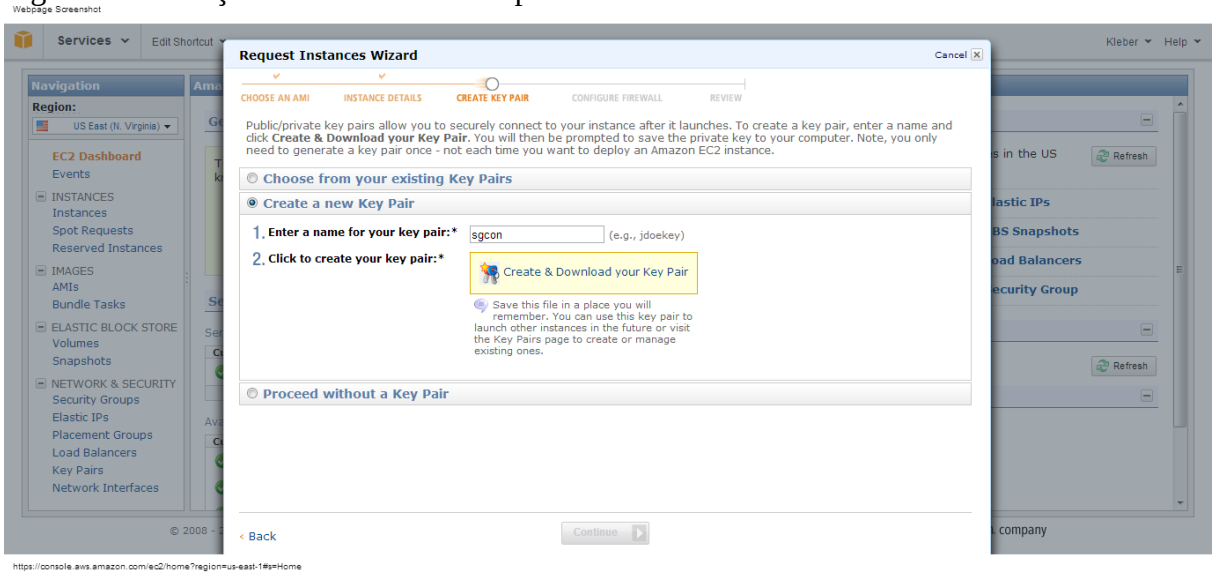
Figura 59 – Criação da instância sétimo passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

O oitavo e próximo passo contempla a tela de criação e download do par de chaves para acesso da instância. Neste passo pode ser usada uma chave já criada anteriormente, criar uma nova ou então continuar sem a criação de um par de chaves, no entanto vale ressaltar que, para esta última opção caso seja perdida a senha de acesso da máquina torna-se impossível o acesso a ela e todos os dados serão perdidos. A figura 60 ilustra a tela de criação e download de chaves privadas.

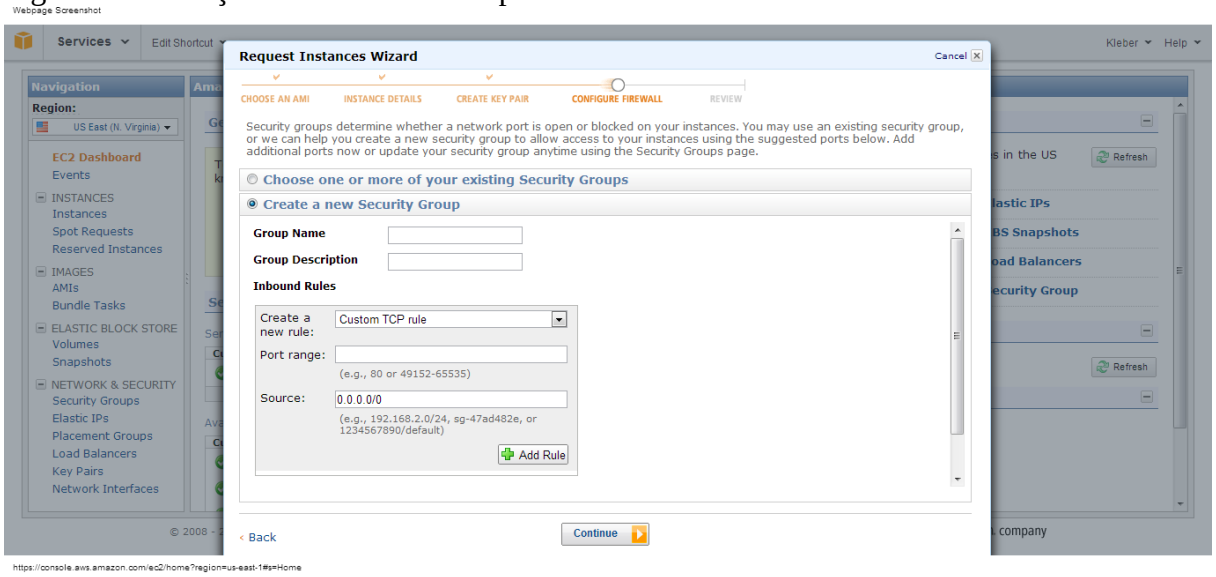
Figura 60 – Criação da instância oitavo passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

Para o nono passo tem-se a definição das regras de *Firewall*, há novamente duas opções de escolha, escolher um grupo de regras anteriormente criada para outras instâncias ou a criação de novas regras. Neste passo poderá ser definido quais as portas de rede ficarão abertas ou bloqueadas para o acesso da instância. A figura 61 abaixo ilustra esta tela.

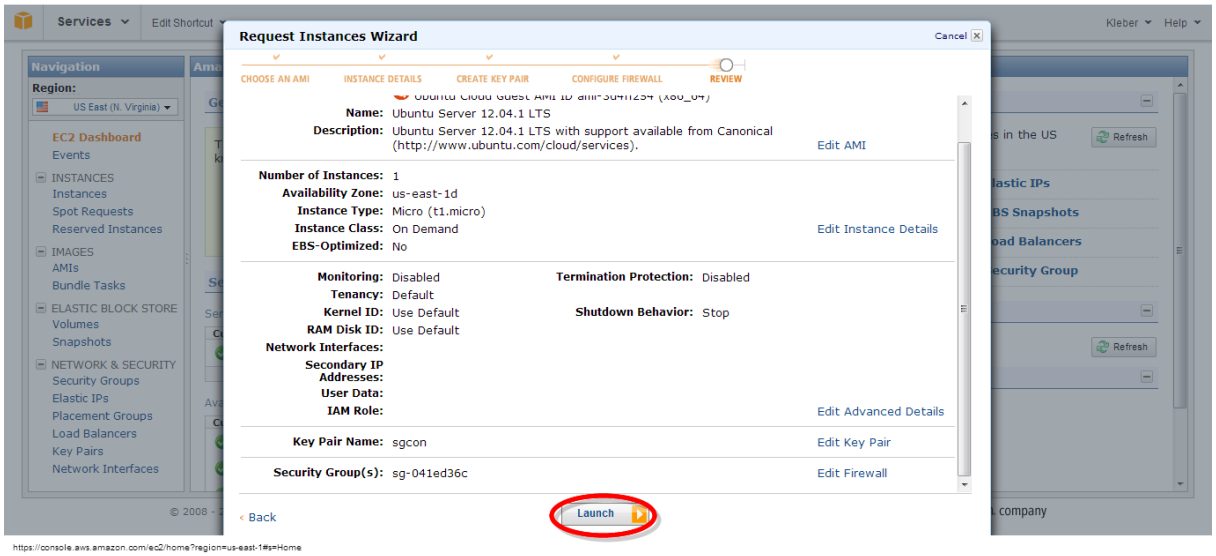
Figura 61 – Criação da instância nono passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

No décimo e último passo tem-se um "*Review"* de todas as configurações antes de confirmar a criação da instância. Caso seja necessário refazer alguma configuração será necessário voltar ao passo que for preciso e ir confirmando até chegar nesta tela novamente. Após a verificação dos dados é necessário selecionar o botão "*Launch*" marcado na figura 62 a seguir.

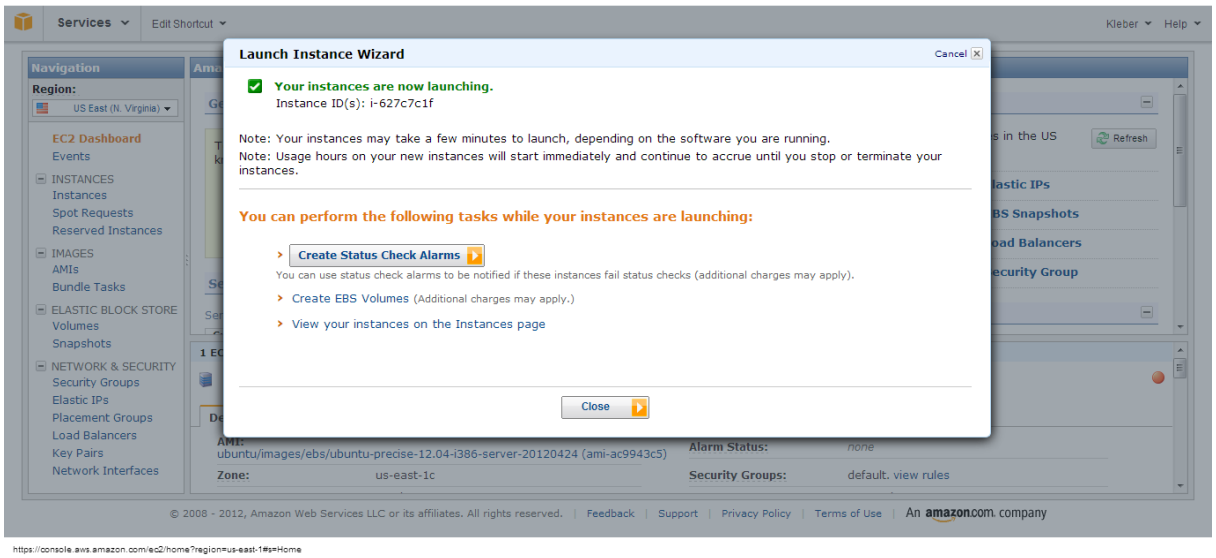
Figura 62 – Criação da instância décimo passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

Após seguir todos os passos anteriores corretamente uma ultima tela é lançada com a confirmação da criação da instância. Esta tela informa que a criação da instância pode levar alguns minutos dependendo do pacote de software escolhido. A mesma tela ainda oferece a criação de alarmes para caso algum problema ocorra com sua instância esses alarmes são disparados e a opção de criar novos volumes para armazenamento de dados. Esta tela pode ser verificada a seguir na figura 63.

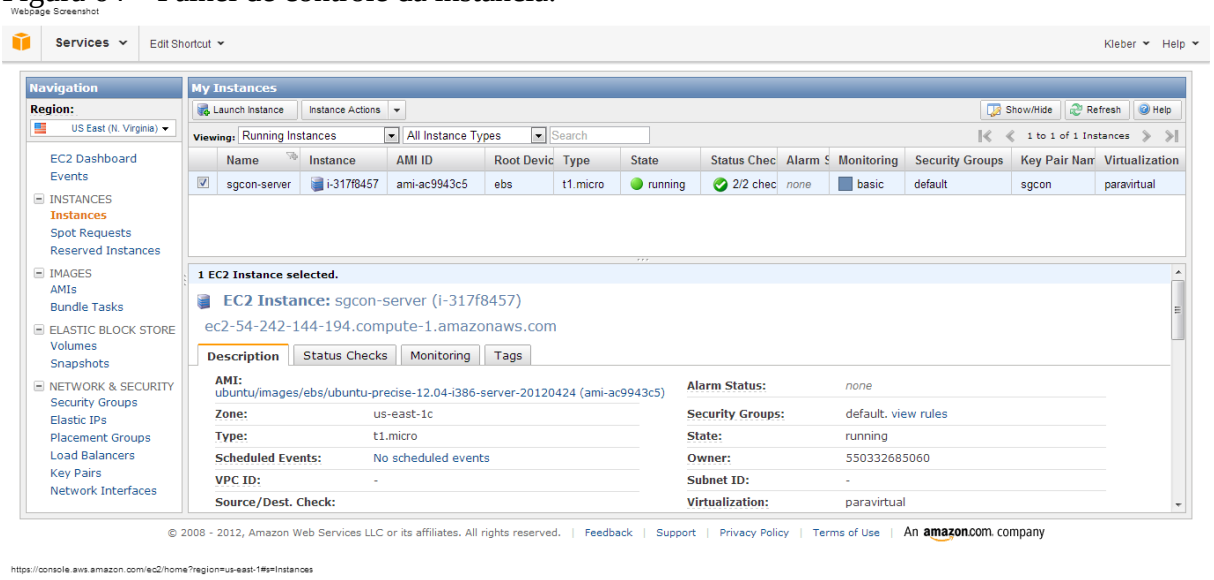
Figura 63 – Criação da instância décimo primeiro passo.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

A próxima tela contempla todas as informações que são pertinentes sobre a(s) instância(s), esta tela disponibiliza um acesso rápido a todas as configurações possíveis para o gerenciamento das mesmas. A figura 64 ilustra esta tela.

Figura 64 – Painel de controle da instância.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

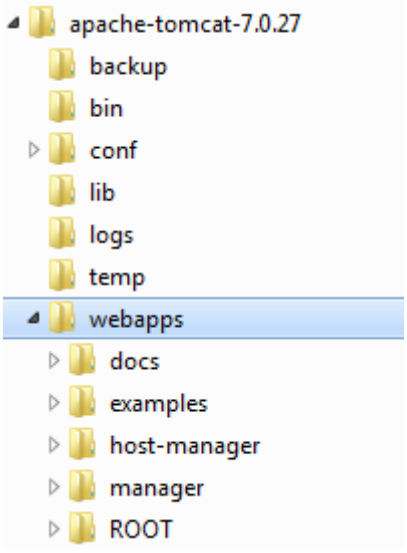
Após a criação da instância foi necessário conectar-se à máquina virtual através de uma conexão *SSH* com a chave sgcon.pem gerada no momento da criação da instância, uma vez conectado na instância foi a necessário prepará-la para receber o aplicativo. Para tanto se fez uso do *Java* 6, *Tomcat7*, e um banco relacional que para este estudo de caso foi o *MySQL*.

A inicialização do servidor web *TomCat* foi feita através de um *script* de inicialização automática juntamente com a instância, desta maneira, sempre que a instância for inicializada o servidor web também será inicializado.

5.1.2.4 *Deploy* da Aplicação

Para cada *deploy* da aplicação no servidor é necessário gerar um arquivo *.war* da aplicação, e através de um protocolo *SCP* enviá-lo para a pasta *webapps* do servidor web conforme pode ser visualizado na figura 65 a seguir.

Figura 65 – Estrutura de pastas do servidor web.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

5.1.2.5 Modificações necessárias

Para a implementação do sistema na nuvem da Amazon não foram necessárias modificações nas tecnologias utilizadas, pois como se trata de uma modalidade IAAS o gerenciamento e implementação das tecnologias utilizadas é de total responsabilidade do administrador da instância. Para o funcionamento da aplicação na nuvem da Amazon as seguintes tecnologias foram mantidas: o Java 6, o servidor web TomCat7 e um banco relacional MySQL, tecnologias exatamente iguais as utilizadas no desenvolvimento em um ambiente tradicional.

5.1.3 Google

O *Google Application Engine* consiste em um conjunto de ferramentas e serviços disponibilizados na web, facilitando a criação de aplicativos que podem ser executados de

forma confiável mesmo sob uma carga pesada e com grandes quantidades de dados. (GOOGLE, 2012).

Incluindo os seguintes recursos:

- Serviço da web dinâmico, com suporte completo a tecnologias da web comuns;
- Armazenamento persistente com consultas, classificação e transações;
- Escalonamento e balanceamento de carga automático;
- APIs para autenticação de usuários e envio de e-mails usando Contas do Google;
- Um ambiente de desenvolvimento local com todos os recursos, simulando o Google App Engine em seu computador;
- Filas de tarefas para realizar trabalho fora do escopo de uma solicitação da web;
- Tarefas programadas para iniciar eventos em horários específicos e em intervalos regulares.

Esse ambiente utiliza os conceitos da nuvem PaaS, de modo que os clientes não se preocupem com os aspectos de configuração e inoperância de servidores, já que estão distribuídos mo provedor do serviço, o qual está configurado de tal modo que a falha de uma de suas máquinas não prejudique o funcionamento do resto do sistema. Sendo baseado em computação em nuvem onde o processamento e o armazenamento do aplicativo são distribuídos entre vários servidores, fazendo necessário ter um ambiente SandBox.

Segundo Müller (2010, p. 33) o SandBox:

> Possibilita que o Google App Engine distribua as solicitações de web da aplicação entre diversos servidores, podendo iniciar ou interromper os servidores para atender às demandas de tráfego e também que cada aplicação possua uma área isolada segura e confiável independente de hardware, sistema operacional e localização física do servidor, garantindo que uma aplicação não influencie no funcionamento das demais aplicações. Este método de virtualização além de possibilitar a distribuição na execução do aplicativo, evita o chamado efeito slashdot, onde em um ambiente compartilhado, o uso abusivo de recursos por uma aplicação afeta o desempenho das demais.

O Google Application Engine disponibiliza suporte a aplicativos criados nas linguagens de programação Python e Java. De modo que para armazenar os dados é utilizado as estrutura de Banco de dados BigTable desenvolvida pela Google. (NUBLING, 2011).

Segundo Chang et al (TRADUÇÃO NOSSA, 2006) o BigTable é

> Um sistema de armazenamento distribuído para gerenciar dados estruturados que é projetado para escalar para um tamanho muito grande: petabytes de dados em milhares de servidores das commodities. Muitos projetos de armazenamento de dados do Google no Bigtable, incluindo indexação web, Google Earth, Google e Finanças. Estas aplicações colocar exigências muito diferentes em Bigtable, tanto em termos de tamanho de dados (a partir de URLs para páginas da web para imagens de satélite) e requisitos de latência (de processamento em massa do servidor a dados em tempo real que servem).

Este serviço é fornecido de modo gratuito, tendo um limite para armazenamento, para o uso da CPU e a largura da banda, de modo que se houver a necessidade de aumentar algum tipo de recuso se tem a possibilidade contratar esses recursos. Para Müller (2010) os recursos oferecidos são suficientes para uma aplicação pequena ou media porte.

Quadro 3 – Valores de hospedagens e de utilização de APIs no Google.

	Quota gratuita por app por dia	**Preço**
On-demand instâncias Frontend	28 horas sem exemplo	0,08 dólares / hora
Instâncias reservados Frontend		0,05 dólares / hora
Datastore replicação alta	1G	$ 0,24 / G / mês
Largura de banda de saída	1G	$ 0,15 / G
Largura de banda de entrada	1G	$ 0.10 / G
Datastore API	50k livre de leitura / gravação / small	$ 0.01/10k ops escrever $ 0.07/10k ler ops $ 0.01/100k ops pequena
Blobstore API	5G	$ 0,17 / G / mês
E-mail API	100 destinatários	Destinatários $ 0,01 / 100
XMPP API	1.000 estrofes	Estrofe $ 0,01 / 1k
Canal API	100 canais abertos	Canais $ 0,01 / 100 abriu
Manipulação de imagem API	✔	✔
Memcache API	✔	✔
Usuários API	✔	✔
Fila de Tarefas	✔	✔
Arquivos API	✔	✔
URL Fetch API	✔	✔
Cron	✔	✔
Prospectivos Search API (experimental)	✔	✔

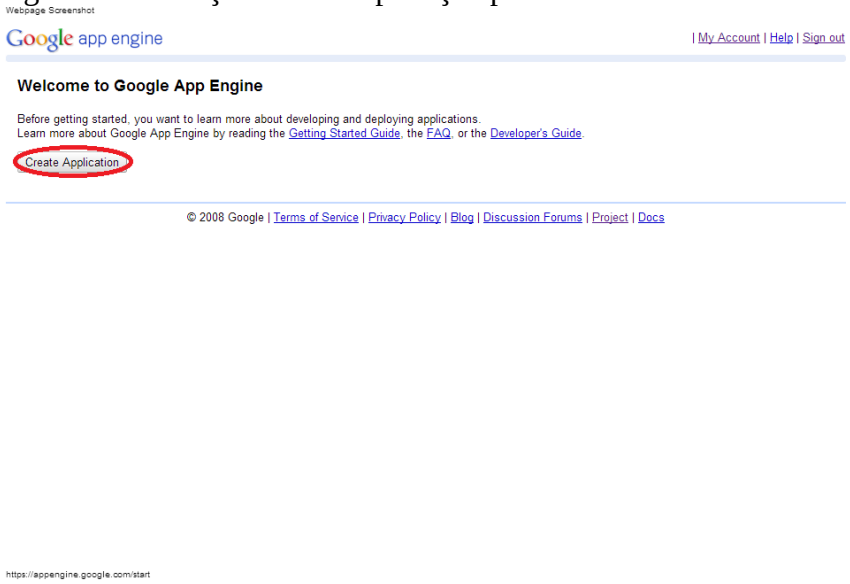
Fonte: (GURGEL, 2011).

A tabela 3 apresenta algumas quotas que a Google disponibiliza para contas gratuitas e os valores cobrados quando as quotas são ultrapassadas.

5.1.3.1 Preparação do ambiente

Para criar uma aplicação em uma nuvem da Google é necessário primeiramente criar uma conta na Google, após a criação é necessário entrar no endereço https://appengine.google.com/ e entrar no sistema com o usuário e senha criados no passo anterior. Após acessar o endereço a tela representada pela figura 66 será apresentada.

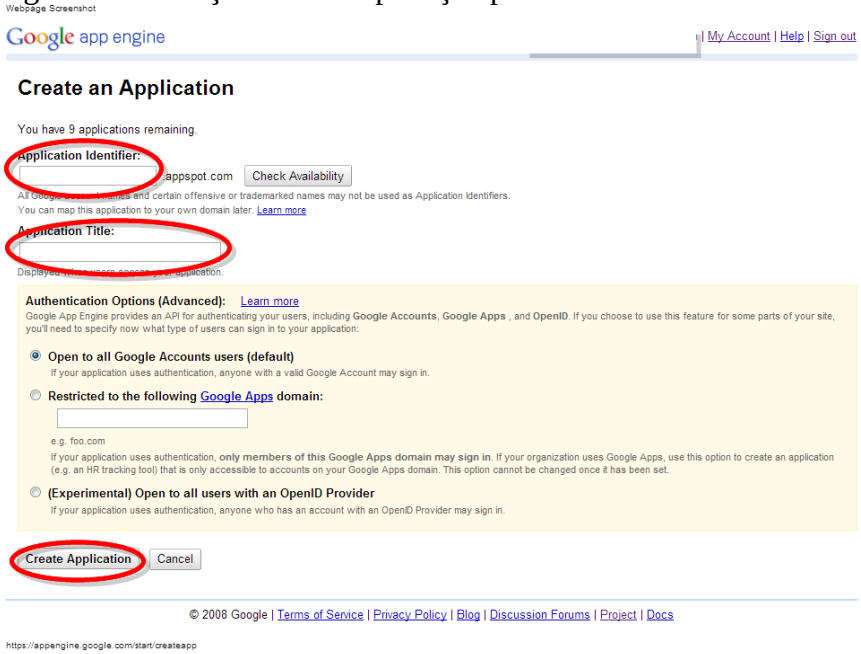
Figura 66 – Criação de uma aplicação passo um.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

O segundo passo é definir o endereço para aplicação e um titulo e selecionar o botão *Create* Application. A figura 67 abaixo representa está tela.

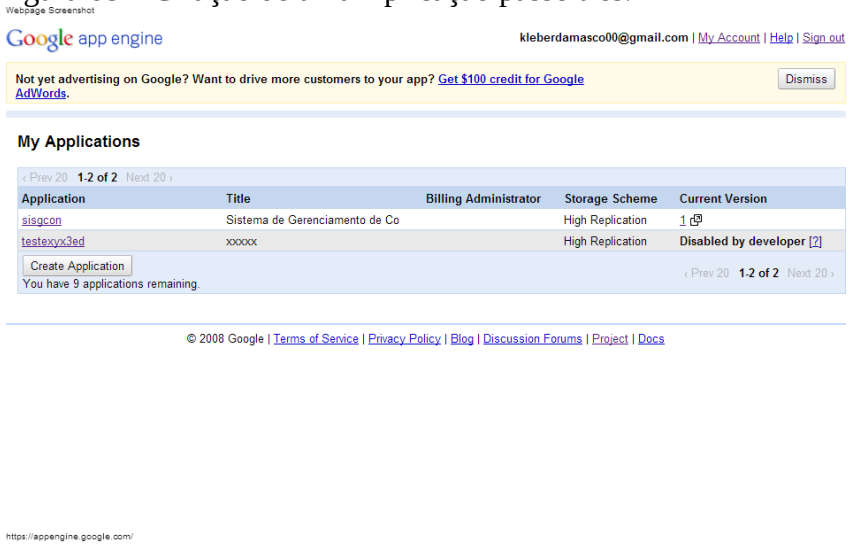
Figura 67 – Criação de uma aplicação passo dois.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

Após a criação da aplicação pode ser observado através da figura 68 logo abaixo todas as aplicações criadas e algumas informações adicionais como o titulo, o administrador do faturamento, o tipo de armazenamento e a versão.

Figura 68 – Criação de uma Aplicação passo três.

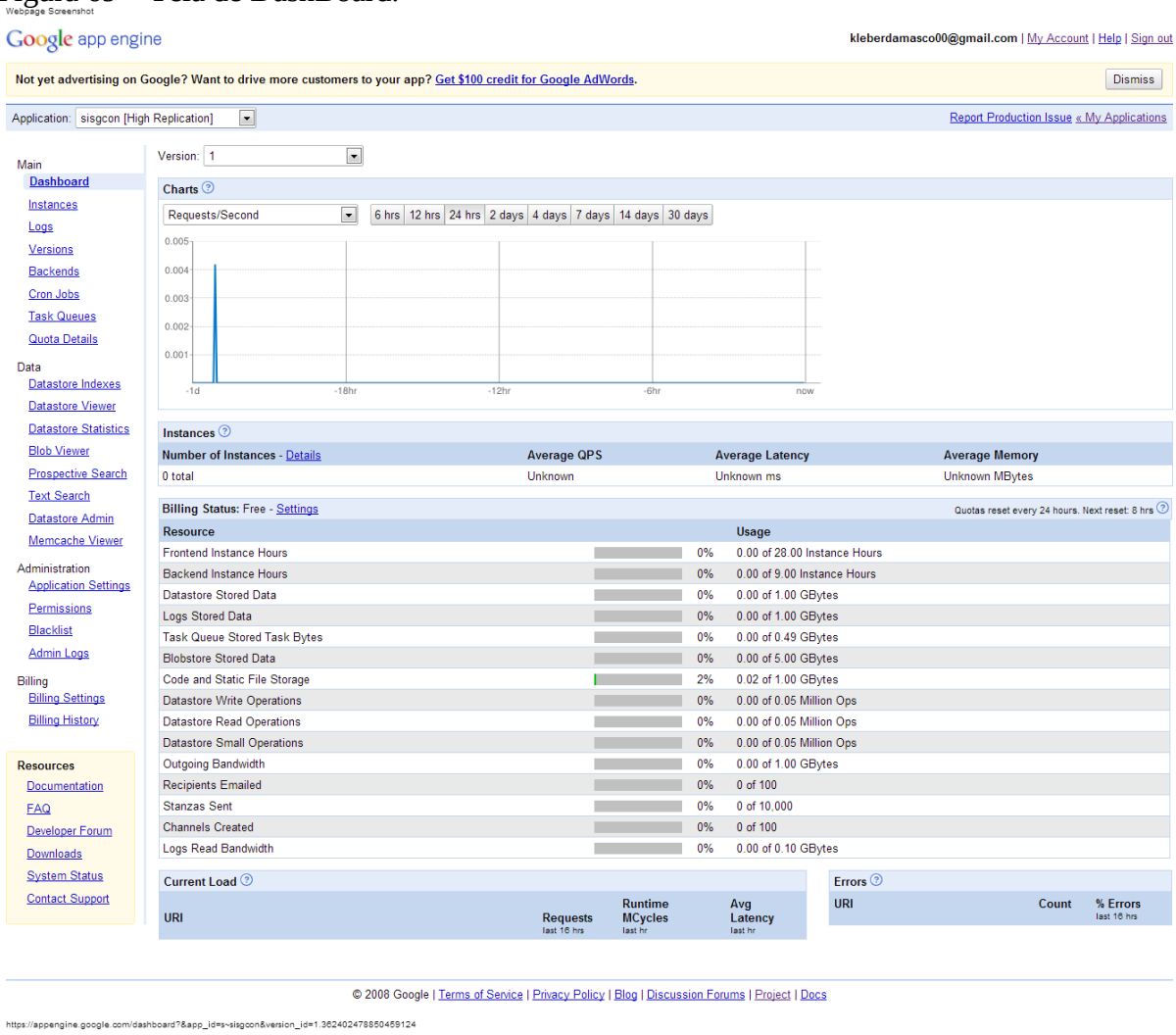


Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

Para o gerenciamento da aplicação e necessário selecionar a aplicação desejada na tela representada pela figura 68 acima, após a seleção será apresentada uma tela chamada de *DashBoard* onde são exibidas todas as informações necessárias e relevantes para a

administração da aplicação, como por exemplo a visualização do *datastore*, *logs* da aplicação, detalhes dos valores gastos, e gerenciamento de performance da aplicação. A figura 69 a seguir contempla esta tela.

Figura 69 – Tela de DashBoard.

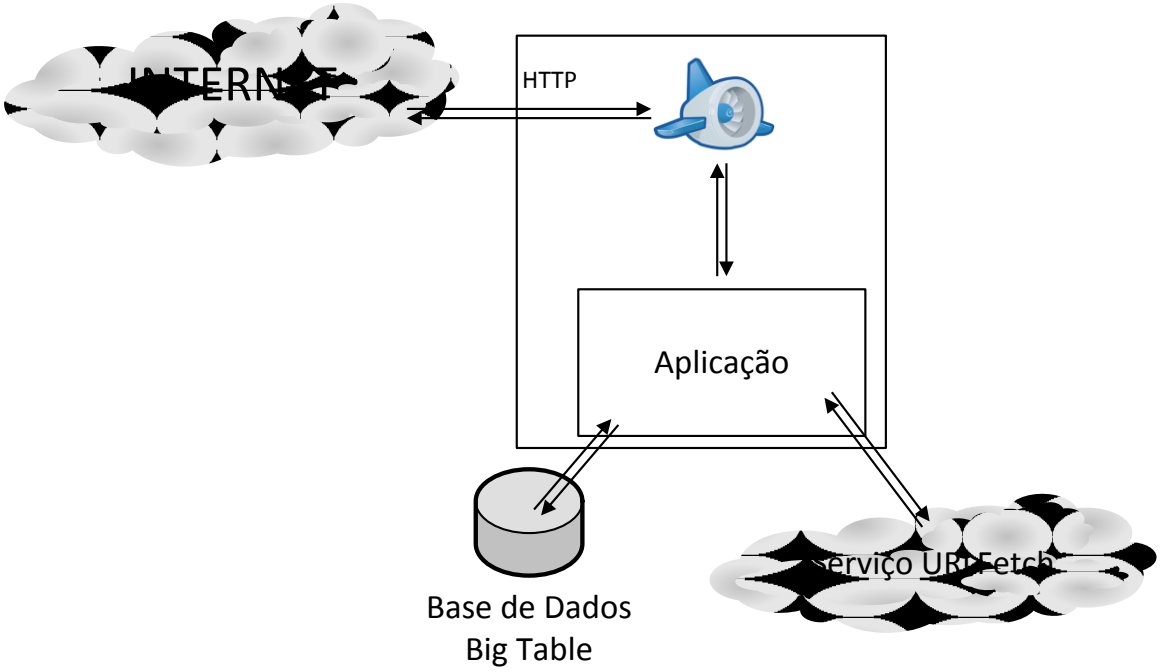


Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

5.1.3.2 Esquema da Solução

Como a Google disponibiliza uma estrutura pronta para que as aplicações possam ser executas e acessadas, na figura 70 será mostrado como é o esquema da Google App.

Figura 70 – Esquema da Solução na Google App

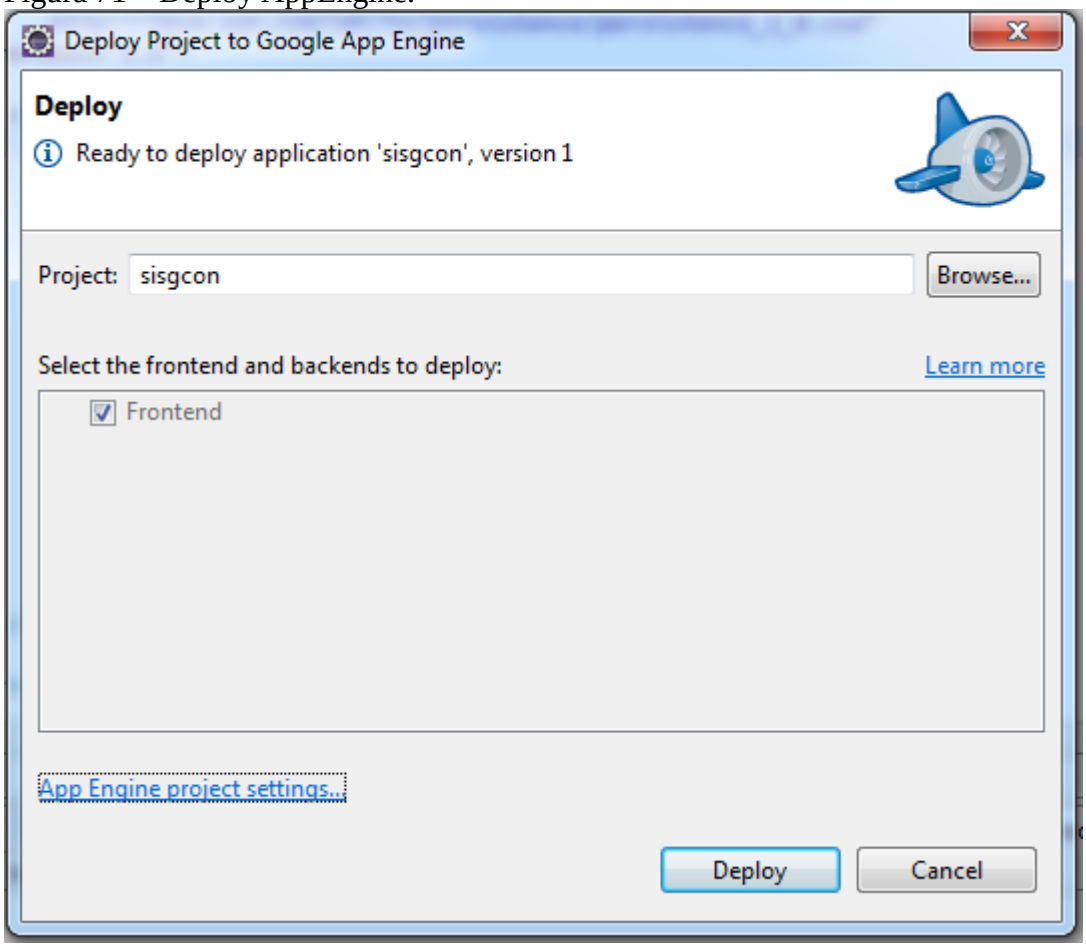


Fonte: (GOOGLE, 2012, ADAPTADO PELOS AUTORES).

5.1.3.3 *Deploy* da Aplicação

Para o *deploy* de uma aplicação em uma nuvem da Google, foi utilizado um *plugin* na IDE Eclipse que faz o serviço de forma simples, rápida e fácil, trazendo desta maneira um atrativo a mais pela facilidade de *deploy* de aplicações em sua nuvem. A figura 71 a seguir ilustra este passo.

Figura 71 – Deploy AppEngine.



Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

Após a execução do procedimento descrito anteriormente a aplicação já estará na nuvem, pronta para ser usada em qualquer lugar e em qualquer dispositivo que possua acesso a internet.

5.1.3.4 Modificações necessárias

O Google App Engine(GAE) estabeleceu uma série de restrições para a utilização de aplicativos na nuvem, por ser um ambiente especificamente limitado e com varias regras e restrições muitas destas implicam diretamente no comportamento tradicional do JSF, desta forma algumas mudanças foram necessárias para o correto funcionamento da aplicação

desenvolvida em JSF2. O GAE exige que o aplicativo execute em uma única *thread*, desta forma se faz necessário a alteração de 2 parâmetros passados no arquivo *web.xml* da aplicação. O primeiro a ser substituído é o parâmetro *com.sun.faces.enableThreading* que deve ser definido como *false*, o segundo a ser definido como *false* e por decorrência do primeiro é o *javax.faces.PROJECT_STAGE*. Para finalizar as configurações é necessário também a adição de dois novos parâmetros, o primeiro é o mecanismo de controle de sessão definida pelo parâmetro *javax.faces.STATE_SAVING_METHOD* que deve ser mantida no lado cliente da aplicação, o último parâmetro a ser definido é o tipo de *expression language* específica do GAE SDK através do parâmetro *com.sun.faces.expressionFactory*, sendo ainda necessário a colocação de um arquivo com a extensão *.jar* chamado de *el-ri-1.0.jar* no *classpath* da aplicação. A figura 72 a seguir contempla esta configuração.

Figura 72 – Configuração web.xml

```
<context-param>
    <param-name>com.sun.faces.enableThreading</param-name>
    <param-value>false</param-value>
</context-param>
<context-param>
    <param-name>javax.faces.PROJECT_STAGE</param-name>
    <param-value>Production</param-value>
</context-param>
<context-param>
    <param-name>javax.faces.STATE_SAVING_METHOD</param-name>
    <param-value>client</param-value>
</context-param>
<context-param>
    <param-name>com.sun.faces.expressionFactory</param-name>
    <param-value>com.sun.el.ExpressionFactoryImpl</param-value>
</context-param>
```

Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

O GAE ainda limita as classes Java permitidas, chamada por eles de *whitelist* disponível no endereço https://developers.google.com/appengine/docs/java/jrewhitelist. Pelo fato do GAE limitar as classes foi necessário clonar a classe Java *com.sun.faces.config.WebConfiguration* o qual o JSF utiliza para realizar algumas configurações *web* já que a mesma continha uma chamada da classe *javax.naming.InitialContext* que não faz parte da *whitelist* definida pelo GAE.

Para a persistência dos dados gerados pela aplicação fora necessário a substituição do provedor *Hibernate* que funciona apenas com bancos de dados relacionais pelo provedor

DataNucleos sob a licença da Apache 2 e que pode persistir os dados em *datastore* baseados em mapas que é o caso da BigTable da Google.

5.2 VALIDAÇÃO DO ESTUDO DE CASO

Para identificar se o processo de adaptação do protótipo para as modalidade PaaS e IaaS, foram conduzidas de maneira correta, levantou-se as principais características de uma aplicação (serviço) de *Cloud Computing*, utilizando-as como base para a validação da migração e adaptação do protótipo. O Quadro 4 apresenta as características e se a migração baseada na modalidade atende ou não.

Quadro 4 – Validação dos ambientes

	GOOGLE APP	**AMAZON EC2**
Atendimento self-service e sob demanda	**Atende**	**Atende**
Elasticidade	**Atende**	**Atende**
Pagamento pelo uso	**Atende**	**Atende**
Garantias de serviço	**Atende**	**Atende**
Acesso ubíquo através da rede.	**Atende**	**Atende**

Fonte: (ELABORAÇÃO DOS AUTORES, 2012).

Para um melhor entendimento da Quadro 4, a seguir será detalhado como os dois modelos propostos atendem as características essenciais desta modalidade, sendo a Google como provedora do serviço PaaS e a Amazon como serviço de IaaS:

- As duas apresentam uma estrutura que garante o atendimento *self-service* e a sob demanda dos recursos, de modo que ambas prestadoras disponibilizam uma estrutura para provisionar os recursos.
- A elasticidade dos recursos computacionais como CPU, largura de banda, memória e entre outros recursos são atendidos pelas duas prestadoras, pois as duas disponibilizam

opções de elasticidades dos recursos computacionais, no período que o serviço necessitar.

- No quesito pagamento pelo uso (*pay-per-use*) há uma diferença entre as duas prestadoras, pois a Google disponibiliza uma quota gratuita, pagando apenas o excedente a essa quota, enquanto a Amazon disponibiliza o serviço gratuitamente por um ano, de modo que após esse período será pago a quantidade de recurso utilizado.
- Tanto a Google quanto a Amazon disponibilizam garantias, através dos acordos de serviços, para a disponibilidade dos serviços.
- As duas possibilitam o acesso ubíquo através da rede, devido ao padrão HTTP para a comunicação, e assim permitindo a comunicação entre diversos tipos de clientes como *browsers*, PDAs, celulares e etc.

Por estas características o serviço portado para as duas estruturas analisadas pode ser classificado como parte de um ambiente de *Cloud Computing.*

5.3 RESULTADOS

Neste tópico são apresentados os resultados obtidos com a criação e adaptações necessárias do protótipo para o correto funcionamento nas duas modalidades propostas por este TCC através de dois subtópicos com nossas percepções e considerações finais sobre a Amazon e o Google.

5.3.1 Amazon

Com o atual preço dos servidores robustos que se tem no mercado, os custos de uma conta de TI que uma empresa mantém como, por exemplo, a manutenção do seu *hardware*, infraestrutura de rede, banco de dados, manter seus técnicos especializados e

atualizados em TI, problemas de disponibilidade 24 x 7, segurança, resfriamento dos equipamentos de processamento de dados, backup´s e no-breaks, a utilização desses serviços em nuvem é uma opção que deve ser levada em consideração. Levando em consideração que o valor para uma hospedagem em nuvem completa ser relativamente baixo, possuir um banco de dados escalável e funcionando juntamente com a aplicação no modelo 24 x 7, mostra que desenvolver em nuvem é a nova forma de ampliar a TI e que ganha espaço a todo o tempo.

A criação de uma instância em um servidor da Amazon através de sua amigável interface trás simplicidade e agilidade para a criação, em poucos minutos a máquina já esta definida e pronta para ser utilizada e escalonada conforme a necessidade. Na Amazon não há limitações com tecnologias a serem utilizadas, a máquina é de total responsabilidade do criador. Um problema identificado neste tipo de ambiente é que sendo de total responsabilidade de um administrador faz-se necessário a manutenção notória exatamente igual a uma máquina local, exemplos como a atualização do sistema operacional e softwares utilizados. Outro ponto fraco está no deploy da aplicação que é relativamente mais demorado em comparação ao ambiente da Google, visto que é necessário conectar-se a instância como administrador e fazer a transferência da aplicação através do protocolo SSH.

Um aspecto importante a ser levantado é sobre a documentação que a Amazon e a Google fornecem, atendendo totalmente o esperado, foi simples iniciar através de vários exemplos desde a criação da instância e suas configurações ou no caso da Google que fornecesse exemplos de classes para a persistência dos dados gerados pela aplicação. Além de fornecerem uma documentação completa é possível também obter ajuda através de fóruns que elas mantêm para auxiliar em dúvidas no qual seja necessário obter-se respostas mais pontuais.

5.3.2 Google

A Google situa-se no mercado tanto na modalidade de SaaS, com todas as suas soluções de aplicações de Office como o Google Agenda, Gmail, Drive entre outros, ou como fornecedor de serviços de PaaS, através do Google App Engine(GAE). De utilização gratuita, até um determinado nível de pedidos, o GAE é uma plataforma de certa maneira limitada na

medida em que apenas suporta aplicações em Python e em Java e o controle sobre o ambiente é algo extremamente limitado. Ao mesmo tempo, as aplicações devem ser desenvolvidas especificamente para esse escopo ou terão de passar por processos de adaptações para poderem ser executadas na plataforma disponibilizada.

Devido as limitações impostas o protótipo necessitou sofrer alterações porque o GAE não vem preparado para aceitar aplicações JSF mas sim aplicações com paginas estáticas do tipo JSP. A GAE por não possuir uma estrutura com banco de dados relacional também oferece limitações na utilização na API de persistência JPA com limitações de consultas polimórficas no qual não é possível obter uma instância de uma subclasse através da consulta da classe pai, ou consultas de agregação dos tipos *sum,avg,max,min,group*. Caso realmente seja necessário fazer esses tipos de consultas, o GAE disponibiliza a opção de usar uma outra API chamada de JDO onde ambas API'S devem ser implementadas pelo *framework* denominado *DataNucleos* licenciada pela *Apache 2* para a realização de todas as transações com o banco de dados.

Em contrapartida as varias limitações impostas um ponto positivo a ser levantando é que Google disponibiliza uma SDK que facilita todo o processo de desenvolvimento, testes e integração para seu aplicativo. Outra vantagem a ser levada em conta é a questão do *deploy* desta aplicação em seus servidores, onde tudo isso pode ser feito através de um *plugin* para a IDE Eclipse que fornece a opção de um simples comando disponibilizar a aplicação na nuvem.

5.4 CONSIDERAÇÕES FINAIS

Muito ainda há de ser analisado antes de decidir qual modalidade deve ser a melhor escolha, as limitações impostas pela Google impactaram diretamente no protótipo criado, sendo desta maneira um essencial estudo dos impactos gerados para a migração de uma aplicação mais robusta tendo em vista que muitas classes Java usadas habitualmente em um desenvolvimento tradicional teriam que ser readaptadas ou substituídas para o funcionamento neste tipo de ambiente.

Após este estudo de caso e analisando todas as vantagens e desvantagens observadas de cada um, se fosse necessário uma escolha para a implantação da aplicação, ficaríamos com a Amazon, pois além de fornecer todas as vantagens de escalabilidade como a Google, ainda é possível ter o controle total da máquina sem qualquer limitação de uso.

6 CONCLUSÃO E TRABALHOS FUTUROS

Neste capítulo serão realizadas as conclusões obtidas acerca do desenvolvimento deste TCC e sugestões sobre trabalhos futuros que possam contribuir de forma mais robusta para o desenvolvimento do sistema para esse novo modelo computacional.

6.1 CONCLUSÕES

Este trabalho apresentou os conceitos sobre o tema computação nas nuvens, definindo e apresentando vantagens e desvantagens relacionadas as modalidades IaaS e PaaS , assim como, apresentar e definir as principais modalidades atuais que este paradigma oferece, como o IaaS, o PaaS e o SaaS, por meio de pesquisa bibliográfica junto a um estudo de caso.

Ficou evidente, com o desenvolvimento deste TCC, que este tema está em grande ascensão, há muito que ser explorado, como por exemplo, novas finalidades e outros benefícios. Como prova disto, o assunto computação nas nuvens está sendo cada vez mais explorado, pois grandes empresas, como as estudadas neste TCC (Google e Amazon), estão investindo e acreditando neste novo conceito e também outras como a IBM e a JBOSS estão tornando-se adeptas a este novo modelo.

Outra questão a ser analisada é a possibilidade de se ter computadores com baixo custo, se houver um número maior de empresas adeptas a este paradigma os computadores não necessitariam possuir grande capacidade de processamento e armazenamento, uma vez que os dados e os aplicativos mais robustos poderão estar nas nuvens necessitando apenas, de um simples dispositivo com acesso a internet para desfrutar destes dados, gerando assim um grande benefício a usuários e empresas.

As vantagens e desvantagens obtidas através do estudo de caso foram significativas para uma eventual escolha de qual seria o melhor tipo de ambiente para implantar uma aplicação, que para o nosso protótipo ficou definido que o ambiente da Amazon seria o mais cabível, pois, além de fornecer todas as características de um ambiente

nas nuvens prove também o total controle da instância, ou seja, o controle da instância é feita exatamente ao controle de uma máquina local onde o administrador pode instalar e configurar tudo o que for necessário para a execução de um sistema.

Com base no estudo de caso foi possível identificar e responder o que seria necessário para tornar o protótipo capaz de ser usado tanto em uma modalidade IaaS fornecida pela Amazon, quanto na modalidade PaaS fornecida pela Google. Apesar das alterações necessárias o sistema proposto foi capaz de realizar todas as tarefas nos dois tipos de ambientes.

6.2 TRABALHOS FUTUROS

A partir deste TCC, futuramente poderá ser realizado um estudo mais aprofundado a respeito de assuntos não citados neste, como a segurança dos dados na nuvem, a migração dos servidores atuais para servidores nas nuvens, um detalhamento dos custos para se manter um servidor com uma grande capacidade na nuvem.

Um possível aprimoramento do sistema desenvolvido, de forma a deixa-lo mais robusto com novas funcionalidades.

Um estudo das outras possibilidades atuais que temos no mercado, como a modalidade PaaS oferecida pela JBOSS chamada de *OpenShift*, ou então, os serviços oferecidos pela IBM.

REFERÊNCIAS

ALMEIDA, Alexandre de. et al. **Pesquisa e Desenvolvimento em Uml.** Araranguá – SC, 2001. Disponível em: < http://joaomorais.com.br/jm/uploads/Links/uml.pdf>. Acesso em: 23 junho 2012.

ARMBRUST, Michael. et al. **Above the Clouds: A Berkeley View of Cloud Computing.** 2009. Disponível em: < http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf>. Acesso em: 24 Abril 2012.

BACELLAR, Hilário Viana, **Cluster: Computação de Alto Desempenho.** Campinas, SP, 2011. Disponível em: <http://www.ic.unicamp.br/~ducatte/mo401/1s2010/T2/107077-t2.pdf > Acesso em: 30 Abril 2012.

BADGER, Lee. et al. **DRAFT Cloud Computing Synopsis and Recommendations.** 2011. Disponível em: <http://csrc.nist.gov/públications/drafts/800-146/Draft-NIST-SP800-146.pdf >. Acesso em: 14 Abril 2012.

BARROS, Aidil Jesus da Silveira; LEHFELD, Neide Aparecida de Souza. **Fundamentos de Metodologia Científica:** Um guia para a iniciação científica. 2. ed. São Paulo: Makron Books, 2000.

BASANT, Narayan Singh. **Cloud Deployment Models – Private, Community, Public, Hybrid with Examples**. 2011. Disponível em: <http://www.techno-pulse.com/2011/10/cloud-deployment-private-public-example.html>. Acesso em 20 Abril 2012

BOGGS, Wendy. et al. **Matering UML com Rational rose.** Rio de Janeiro - RJ, 2002.

BOLSONI, Evandro Paulo. **Computação Ubíqua, Cloud Computing e PLC para Continuidade Comunicacional diante de Desastres**. 2009. Disponível em: <http://www.defesacivil.uff.br/defencil_5/Artigo_Anais_Eletronicos_Defencil_14.pdf>. Acesso em 19 Abril 2012

BORTOLIN, Elcio Luiz Pagani. **Alta Disponibilidade usando CODA e LVS**. Minas Gerais, 2005.

BURNETTE, Ed. **Eclipse Ide - Usando o Supercompleto Ide - Guia de Bolso**. Editora Bookman, 2006

BUYYA, R. et al. **Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities.** CoRR, (abs/0808.3558), 2008.

CAMARGO JUNIOR, João Bastista. et al. **Sistemas interligados de gestão ERP e cloud computing: características, vantagens e desafio.** UNIMEP, 2010.

CANCIAN, Maiara H. **Uma Proposta De Guia De Referência Para Provedores De Software Como Um Serviço.** 2009. Disponível em: <http://www.das.ufsc.br/~maiara/files/dissert_maiara.pdf>. Acesso em 26 Abril 2012.

CARNEIRA, Ricardo José Gouveia. et al. **A SEGURANÇA NA PRESERVAÇÃO E USO DAS INFORMAÇÕES NA COMPUTAÇÃO NAS NUVENS.** Faculdade de Tecnologia de João Pessoa, (2011). Disponível em: <www.fatecjp.com.br/revista/art-ed02-001.pdf> Acesso em 04 maio 2012.

COSTA, Carlos J. **Desenvolvimento para Web.** Lusocrédito, 2007. Disponível em: <http://books.google.com.br/books?id=Jn6dTDF-wcsC&pg=PT101&dq=DIAGRAMA+DE+CLASSES&hl=pt-BR&sa=X&ei=SgOMUOyCOdC40gH65YGQBA&ved=0CE0Q6AEwBg#v=onepage&q&f=false> Acesso em 28 outubro 2012.

CEARLEY, David W**. Cloud Computing - Key Initiative.** 2010. Disponível em: <http://www.gartner.com/it/initiatives/pdf/KeyInitiativeOverview_CloudComputing.pdf>. Acesso em: 24 mar 2012.

CHANG, Fay. et al. **Bigtable: A Distributed Storage System for Structured Data.** 2006. Disponível em: <http://research.google.com/archive/bigtable.html>. Acesso em: 18 outubro 2012.

CLAIR, Guy ST. **Software-as-a-Service (SaaS)Put the Focus on theKM/Knowledge Services Core Function.** 2008. Disponível em < http://smr-knowledge.com/wp-content/uploads/2010/01/EOS-SaaS-White-Paper-2008.pdf>. Acesso em: 27 Abril 2012.

CONTI, Fabieli De. **Grades Computacionais para Processamento de Alto Desempenho.** 2008. Disponível em: <www-usr.inf.ufsm.br/~andrea/elc888/artigos/artigo3.pdf>. Acesso em: 03 maio 2012.

DANTAS, Mario, **Computação Distribuída de Alto Desempenho: Redes, Clusters e GridsComputacionais**. 2005.

ENGHOLM JR., HÉLIO. **Engenharia de Software na prática.** Editora Novatec, São Paulo, 2010.

ERL, Thomas. **SOA Princípios de design de serviços.** São Paulo, 2008.

FERMO. Software as a Service. Disponível em:<http://www.fermo.com.br/images/saas.png>. Acesso em: 22 abril 2012.

FOSTER, Ian. et al. **Cloud Computing and Grid Computing 360-Degree Compared.** USA, 2008 Disponível em:< http://arxiv.org/abs/0901.0131>. Acesso em: 28 de Abril 2012.

FOWLER, Martin. UML essencial: um breve guia para a linguagem-padrão de modelagem de objetos. Trad. João Tortello, 3 ed. Porto Alegre: Bookman, 2005.

FRONDANA, Ten GIOVANI. et al. **Arquitetura Orientada a Serviços para Gestão de Processos Acadêmicos na Web,** Rio de Janeiro, 2009. Disponível em: <*www.comp.ime.eb.br/.../PFC-Arq_**Gestao**_Proc_Acad_**WEB**.pdf*>. Acesso em 3 de Agosto 2012.

FURLAN, José Davi. **Modelagem de objetos através do uml.** Makron books, 1998.

GARTNER Group. **Gartner Highlights Five Attributes of Cloud Computing.** 2009. Disponível em: < http://www.gartner.com/it/page.jsp?id=1035013> Acesso em 04 maio 2012.

GIL, Antônio C. **Como elaborar projetos de pesquisa.** 4. ed. São Paulo: Atlas, 2002.

GOOGLE. Google Developers, 2012. Disponível em: <https://developers.google.com/appengine/docs/whatisgoogleappengine?hl=pt-br>. Acesso em: 8 outubro 2012.

GURGEL, Diego Oliveira. **CLOUD COMPUTING.** FACTHUS – UBERABA, 2011. Disponível em: <http://pt.scribd.com/doc/86758029/TCC1-1-Diego-7#outer_page_33>. Acesso em: 19 outubro 2012.

HAYES, Brian. **Cloud Computing.** Communications of the ACM, Vol.51, n.7, 2009.

JANDL, Peter Jr. **Introdução ao Java.** Universidade São Francisco, 1990.

JUNIOR, L. Gilmar. et al. **Uma aplicação de Gestão Acadêmica Utilizando Cloud Computing.** Gravataí-RS, 2010.

KAUFMAN, L. M. **Data Security in the World of Cloud Computing**. IEEE Security and Privacy, 7(4): 61-64, Jul 2009.

K19, Treinamentos. **Desenvolvimento Web Com Jsf2 e Jpa2**. 2012. Disponível em: <http://pt.scribd.com/doc/101634145/Desenvolvimento-Web-Com-Jsf2-e-Jpa2#outer_page_14>. Acessado em: 02 outubro 2012.

LAWTON, G. **Developing Software Online With Platform-as-a-Service Technology** Computer, vol. 41, no. 6, pp. 13-15, Jun 2008, doi:10.1109/MC.2008.185

LEAVITT, Neal. **Is Cloud Computing Really Ready for Prime Time?** 2009**.** Disponível em: <http://www.leavcom.com/ieee_jan09.htm>. Acesso em: 21 Abril 2012.

LEINER, Barry M. et al. **Brief History of the Internet**. Disponível em: <http://www.internetsociety.org/internet/internet-51/history-internet/brief-history-internet>. Acesso em: 24 mar. 2012.

LONGO, Fernando. et al. **Engenharia de Requisitos.** 2005. Disponível em: <http://www.inf.ufsc.br/~wesley/engSoft/> Acessado em: 25 junho 2012.

MAIA, José Anízio. **Construindo softwares com qualidade e rapidez usando ICONIX**, 2005. Disponível em: <http://www.guj.com.br/content/articles/patterns/iconix_guj.pdf> Acesso em: 16 julho 2012.

MARTINS, Adriano. **Fundamentos de Computação Nuvem para Governos.** Brasilia-DF, 2010.

MELL, Peter. GRANCE, Timothy. **The NIST Definition of Cloud Computing**, 2009**.** Disponível em: < http://www.nist.gov/customcf/get_pdf.cfm?pub_id=909616>. Acesso em: 21 Abril 2012.

MENDES, Antônio. **Arquitetura de software: desenvolvimento orientado para arquitetura.** Rio de Janeiro, 2002.

MENEZES, Estera Muszkat; SILVA, Edna Lúcia da. **Metodologia da pesquisa e elaboração de dissertação.** 4. ed. ver. atual. Florianópolis: UFSC, 2005.

MEYER, Bertrand. et al. Desenvolvimento **Orientado a Objetos: o método fusion.** Rio de Janeiro: Campus, 1996.

MILANI, André. **MySQL - Guia do Programador**. Editora NovaTec, 2007.

NUBLING, Gabriela. **Cloud Computing aplicada ao Cenário Corporativo.** FATEC – São Paulo, 2011.

MÜLLER, Victor Daniel. **Desenvolvimento de aplicações sob o paradigma da computação em nuvem com ferramentas Google.** UFSC, 2010.

NAITO, Daniel da Silva. et al. **Utilização De Banco De Dados Em Computação Nas Nuvens**. São José Dos Campos, 2010.

ORACLE. Oracle Corporation, 2012. Disponível em: <http://www.oracle.com/technetwork/java/javase/jdbc/index.html>. Acessado em: 15 outubro 2012.

NEVES, Pedro. et al. **O GUIA PRÁTICO DO MySQL**. Editora Centro Atlântico, 2005.

PITANGA, Talita. **JavaServer Faces: A mais novatecnologia Java para desenvolvimento WEB**. 2011. Disponível em: <http://pt.scribd.com/doc/72370958/jsf>. Acessado em: 02 outubro 2012.

PAES, Evandro. **Introdução à java persistence api – JPA**, 2007. Disponível em: < http://evandropaes.wordpress.com/2007/06/22/introducao-a-java-persistence-api-%E2%80%93-jpa/>. Acessado em: 10 outubro 2012.

POSSOBOM, Camila Cerezer. **Estudo De Caso: Cloud Computing - Computação Em Nuvem**. Ijuí - RS, 2010.

RAUEN, Fábio José. **Roteiros de investigação científica.** Tubarão (S.C.): UNISUL, 2002.

ROCHA, Gabriel. et al. **Camada de Persistência de Dados para Aplicações Java: O Hibernate**. Disponível em: <http://pt.scribd.com/doc/62897364/Conceitos-Hibernate>. Acesso em: 0l outubro 2012.

ROSENBERG, Doug. et al. **Agile Development with ICONIX Process: People, Process, and Pragmatism**. New York: Apress, 2005.

RUIZ, João Álvaro. **Metodologia Científica:** Guia para eficiência nos estudos. São Paulo: Atlas, 1982.

SALVADOR, Fábio Burch. LOUCADEMIA DE JAVA, 2008. Disponível em: <http://www.guardian.co.uk/technology/2008/apr/17/google.software>. Acesso em: 10 outubro 2012.

SCHOFIELD, J. **Google angles for business users with 'platform as a service'**, 2008. Disponível em: <http://www.fabiosalvador.com.br/apostilas/apostilajava.pdf>. Acesso em: 27 Abril 2012.

SCHULLER, Sinclair. **Demystifying the cloud: Where do SaaS, PaaS and other acronyms fit in?**. 2008. Disponível em: <http://www.saasblogs.com/saas/demystifying-the-cloud-where-do-saas-paas-and-other-acronyms-fit-in>. Acesso em: 20 abril 2012.

SILVA, Alberto Manuel Rodrigues da, et al**. UML, Metodologias e Ferramentas CASE**. Edições Centro Atlântico. Portugal, 2001.

SILVA, George, et. al. **Utilizando Iconix no Desenvolvimento de Aplicações Delphi.** João Pessoa - PB – 2007. . Disponevel em: <*http://www.redenet.edu.br/públicacoes/arquivos/20080212_080829_INFO-067.pdf*>. Acesso em: 24 junho 2012.

SOARES, Robson dos Santos. **Sistemas Distribuídos.** Faculdade 7 de Setembro, 2011. Disponível em: < http://pt.scribd.com/doc/60555610/Computacao-Nuvem>. Acesso em: 01 maio 2012.

SOUSA, Flávio R. C. et al. **Computação em Nuvem: Conceitos, Tecnologias, Aplicações e Desafios.** Universidade Federal do Ceará, 2009

TAGLIRI, Fabiane, et al. **GRIDS COMPUTACIONAIS.** Parana, 2006. Disponevel em: <*renderfarm.googlecode.com/files/305-979-1-PB.pdf* >. Acesso em: 30 Abril 2012.

TAURION, Cezar**. SOA: dando os primeiros passos.** 2007. Disponível em: <*www.cin.ufpe.br/~ajsc2/**SOA**%20e%20os%20**primeiros**%20passis.pdf*>. Acesso em: 13 agosto 2012.

TAURION, Cezar. **Cloud Computing: computação em nuvem: transformando o mundo da tecnologia da informação.** Rio de Janeiro, 2009.

TECHMIXE. **Cloud Computing - Emerging Computing Technology**. 2010. Disponível em: <http://www.techmixer.com/cloud-computing-the-emerging-computing-technology>. Acesso em:: 14 abril 2012.

TEIXEIRA, Rafael Augusto. **SW-V: Modelo de streaming de software baseado em técnicas de virtualização e transporte peer-to-peer.** UNESP, 2010.

VECCHIOLA, Christian. et al. **Aneka: A software platform for .NET-based Cloud Computing.** In W. Gentzsch, L. Grandinetti, G. Joubert(Eds). High Speed and Large Scale Scientific Computing. IOS Press, Amsterdam, Netherlands, (2009).

VERDI, Fábio Luciano. et al . **"Novas Arquiteturas de Data Center para Cloud Computing"** in: mini-curso do Simpósio Brasileiro de 28 Redes de Computadores e Distribuídos siste-mas (SBRC 2010), Gramado, 2010.

WINDOWS AZURE. Disponível em: <http://www.windowsazure.com/pt-br/home/features/overview/>. Acesso em: 28 Abril 2012.