WESLEY GOMES DA SILVA

ARQUITETURA ORIENTADA A SERVIÇOS DESTINADO A CONTRATAÇÃO DE PRESTADORES DE SERVIÇOS AUTÔNOMOS

Palmas – TO

2014

Wesley Gomes da Silva

ARQUITETURA ORIENTADA A SERVIÇOS DESTINADO A CONTRATAÇÃO DE PRESTADORES DE SERVIÇOS AUTÔNOMOS

Trabalho de Conclusão do Curso de Sistemas de Informação da Faculdade Católica do Tocantins, apresentado com parte dos requisitos para obtenção do título de Bacharel em Sistemas de Informação.

Orientador: M. Sc. Marco Antônio Firmino de Sousa.

Palmas – TO

2014

Wesley Gomes da Silva

ARQUITETURA ORIENTADA A SERVIÇOS DESTINADO A CONTRATAÇÃO DE PRESTADORES DE SERVIÇOS AUTÔNOMOS

Esta monografia foi julgada adequada pra obtenção do diploma de Bacharel em Sistemas de Informação do curso de graduação em Sistemas de Informação da Faculdade Católica do Tocantins.

Banca Examinadora

Assinatura do Orientador

Membros da Banca Examinadora

_____,_____/_____/_____

Local e data de aprovação:

Nota: _____

AGRADECIMENTOS

Agradeço primeiramente a Deus o meu grande guia, por ter me abençoado com a chance de cursar em uma grande faculdade como a CATÓLICA DO TOCANTINS - FACTO e ter recebido subsídios tão ricos durante essa jornada.

Ao professor Marco Antônio Firmino de Sousa pela orientação e motivação para realização deste trabalho, e por exercer tão bem o dom de repassar seus conhecimentos além de ser uma referência aos alunos.

Agradeço em especial a minha família, pai, mãe e irmã, por estar ao meu lado na busca por este sonho e também aos meus amigos que souberam conviver e respeitar ainda que nem sempre compartilhássemos as mesmas ideias. E por tudo, a saudade há de ficar.

Aos meus professores que durante esse tempo que passou transmitiram não só conhecimento teórico, mas lições de vida me engrandecendo como pessoa e profissional.

A todos, agradeço por tudo e muito obrigado!

Dedico este trabalho ao Deus
eterno e toda a minha família

RESUMO

A adoção de Arquitetura Orientada a Serviços - SOA tem se tornado prática frequente em organizações que buscam resolver problemas de flexibilidade, mudanças e de integração entre suas aplicações, SOA chega com o objetivo de compartilhar informações entre aplicações distintas através de serviços bem definidos e trazendo consigo vantagens como reutilização dos serviços criados em outras aplicações, facilidade de manutenção dos serviços, integração com outros serviços, assim disponibilizando uma padronização. O objetivo deste trabalho foi criar um mecanismo para o controle de prestadores de serviços através da aplicação de SOA, introduzindo uma nova visão na concepção de desenvolvimento de aplicações web. Em paralelo a aplicação de SOA no trabalho, foi desenvolvido um *Web Service* baseado nos protocolos SOAP, UDDI, WSDL e XML que implementam interações remotas entre os aplicativos sob redes baseadas no protocolo IP, garantindo assim o armazenamento dos serviços disponibilizados pelo projeto e disponibilizando-os para consumo das diversas aplicações clientes.

Palavra-chave: Arquitetura Orientada a Serviço, Web Service, Busca de Serviços.

ABSTRACT

The adoption of Service Oriented Architecture - SOA has become common practice for organizations seeking to resolve problems of flexibility, change and integration among its applications, SOA's objective is sharing information between different applications through well defined services, bringing advantages such as: reuse of services that were created in other applications, ease of maintenance, integration with other services, which also provides a standard. The aim was to create a mechanism to control service providers through SOA by introducing a new vision in designing web application development. In addition to applying SOA in this project, we developed a Web Service based on SOAP, UDDI, WSDL and XML protocols that implement remote interactions between applications in IP-based networks, ensuring the services storage provided by the project and making it available to the consumption of the various customer applications.

Keywords: Service Oriented Architecture, Web Service, Search Services.

ÍNDICE DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

API - Application Programming Interface

CEO - Executive Officer

FACTO - Faculdade Católica do Tocantins

FIETO - Federação das Indústrias do Estado do Tocantins

HTTP - Hypertext Transfer Protocol

IBGE - Instituto Brasileiro de Geografia e Estatística

IDE - Integrated Development Environment

MVC - Model View Controller

NTI - Núcleo de Tecnologia da Informação

PIB - Produto interno bruto

SGBD - Sistema de Gerenciamento de Banco de Dados

SINE - Sistema Nacional de Emprego

SOA - Arquitetura Orientada a Serviços

SOAP - Simple Object Access Protocol

TI - Tecnologia da Informação

UDDI - Universal Description, Discovery and Integration

UML - Unified Modeling Language

WSDL - Web Services Description Language

XHTML - eXtensible Hypertext Markup Language

XML - eXtensible Markup Language

ÍNDICE DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

As famosas páginas amarelas ajudaram muita gente a encontrar os mais diversos serviços no passado, de pintores e eletricistas a professores particulares e detetives. Hoje quem cumpre esse papel é a internet. Mas como saber se os serviços anunciados são mesmo confiáveis ou de qualidade?

Pensando em resolver este problema, este trabalho propõe o desenvolvimento de uma ferramenta que ajuda o consumidor na hora de encontrar e avaliar diferentes serviços sistematizando todo um processo de contratação de um profissional.

O desafio de inovar em serviços seja nas empresas que os prestam (seguradoras, empresas de tecnologia da informação, laboratórios de análises clínicas, escolas, circos); seja em uma indústria que faz pós-venda ou na oferta de serviços complementares a produtos, a realidade é que a renda gerada em serviços há muito superou a dos produtos em uma economia como a brasileira. Segundo o Instituto Brasileiro de Geografia e Estatística (IBGE), em 2012, a atividade de serviços respondeu por 68,5% do Produto Interno Bruto (PIB) e proporcionou mais de 78% dos empregos formais do país.

O objetivo do sistema é ser uma *marketplace* para prestadores de serviços, que facilite a vida de quem está procurando profissionais de qualidade, bons preços e satisfazer uma determinada necessidade ou desejo de um cliente, poupando tempo na contração de um serviço.

A principal finalidade dessa aplicação é executar funções que, caso um ser humano fosse executar, seriam consideradas inteligentes. Podemos pensar em algumas características básicas desse sistema, como a capacidade de raciocínio (aplicar regras lógicas a um conjunto de dados disponíveis para chegar a uma conclusão), aprendizagem (aprender com os erros e acertos de forma que futuramente possa agir de maneira mais eficaz), reconhecer padrões (padrões de comportamento) e inferência (capacidade de conseguir aplicar o raciocínio nas situações do nosso cotidiano). De acordo com (LUGER, 2004), um sistema que possua uma ou mais dessas características pode ser considerado um sistema inteligente.

Espera-se para esse sistema que ele possua capacidade de realizar associações de forma que a busca seja a mais adequada à necessidade de quem pesquisa o serviço. Estas

associações possibilitarão que os usuários encontrem o serviço não apenas pelo termo da busca, mas por termos equivalentes como: regionalismos e sinônimos.

1.1. DEFINIÇÃO DO PROBLEMA

O problema consiste em desenvolver um ambiente integrado que disponibilize de maneira transparente informações de prestadores de serviços a possíveis clientes. Independentemente da plataforma utilizada, o acesso aos dados dos prestadores de serviços podem ser consumido pelos clientes. Sistemas como GetNinjas, Iguanafix e Recomind.net são capazes de fornecer informações úteis e em tempo real sobre prestadores de serviços, serviços ofertados, qualificações de serviços e localização, porém são apenas aplicações web ou móvel isolada. A transparência no acesso às informações permite a criação de um ecossistema homogêneo poderoso capaz fornecer e suportar integrações a diversas aplicações. Além disso, os sistemas supracitados são soluções regionalizadas com atuação limitada às regiões sul e sudeste, com isso a proposta desse trabalho busca solucionar um problema regional em específico no estado do Tocantins.

1.2. JUSTIFICATIVA

Aplicações inteligentes que nos forneçam informações precisas, rápidas são indispensáveis devido a gestão do tempo cada vez mais difícil, a criação ou melhoramento de ferramentas que nos auxiliem no meio pessoal ou profissional far-se-ão necessárias.

1.3. OBJETIVO

Criar uma arquitetura orientada a serviços destinado a contratação de prestadores de serviços autônomos.

1.3.1 OBJETIVOS ESPECÍFICOS

- Criar o banco de dados para o armazenamento e consumo das informações;
- Modelar a ferramenta proposta;
- Criar serviços que possam ser consumidos por diferentes plataformas;
- Desenvolver uma aplicação de teste com o objetivo de demonstrar o consumo dos serviços desenvolvidos.

1.4. MOTIVAÇÃO

Melhorar a vida das pessoas com o auxílio da tecnologia, tornando mais fácil a contratação de serviços, informar prestadores de serviços mais próximos de sua localização, visualização de comentários referentes aos serviços prestados, auxílio na pesquisa dos serviços. Enfim uma ferramenta que auxiliem na vida das pessoas dispondo de informações úteis e inteligentes de maneira fácil e rápida.

1.5. ESTRUTURA DA MONOGRAFIA

A seção 2(dois) desse trabalho mostra a pesquisa bibliográfica utilizada no trabalho. A seção 3(três) mostra o referencial teórico que aborda os principais conceitos que envolvem a utilização de SOA e trabalhos relacionados. A metodologia será apresentada na seção 4(quatro), descrevendo os métodos e procedimentos utilizados neste trabalho. A visão do sistema e a codificação serão expostas na seção 5(cinco) apresentando com detalhes o processo de construção dos serviços, bem como os principais trechos de código utilizados no processo. E, por fim, a seção 6(seis) apresenta informações sobre as conclusões e os trabalhos futuros.

2 PESQUISA BIBLIOGRÁFICA

O seguimento das micro e pequenas empresas brasileira encontram-se em expansão, impulsionado pelo crescimento de crédito e renda. Anualmente, no país são criados mais de 1,2 milhões de novos empreendimentos formais, sendo que desse total 99% correspondem à micro e pequenas empresas e empreendedor individual (SEBRAE, 2013). A maior parte dos negócios criados no Brasil é concebida por pequenos empresários, que nem sempre possuem conhecimento de gestão de negócios, e acabam atuando de forma empírica e sem planejamento (DORNELAS, 2005).

Neste contexto, segundo Osterwalder & Pgneur (2011), a aplicação da ferramenta Bussines Model Canvas é uma forma prática de analisar e refletir sobre o funcionamento de um Modelo de Negócio e auxilia no direcionamento da empresa, na definição de planos de ação no que se refere aos processos, estrutura e sistemas organizacionais. Sendo assim, a Figura **1** apresenta o Modelo de Negócio do projeto proposto.

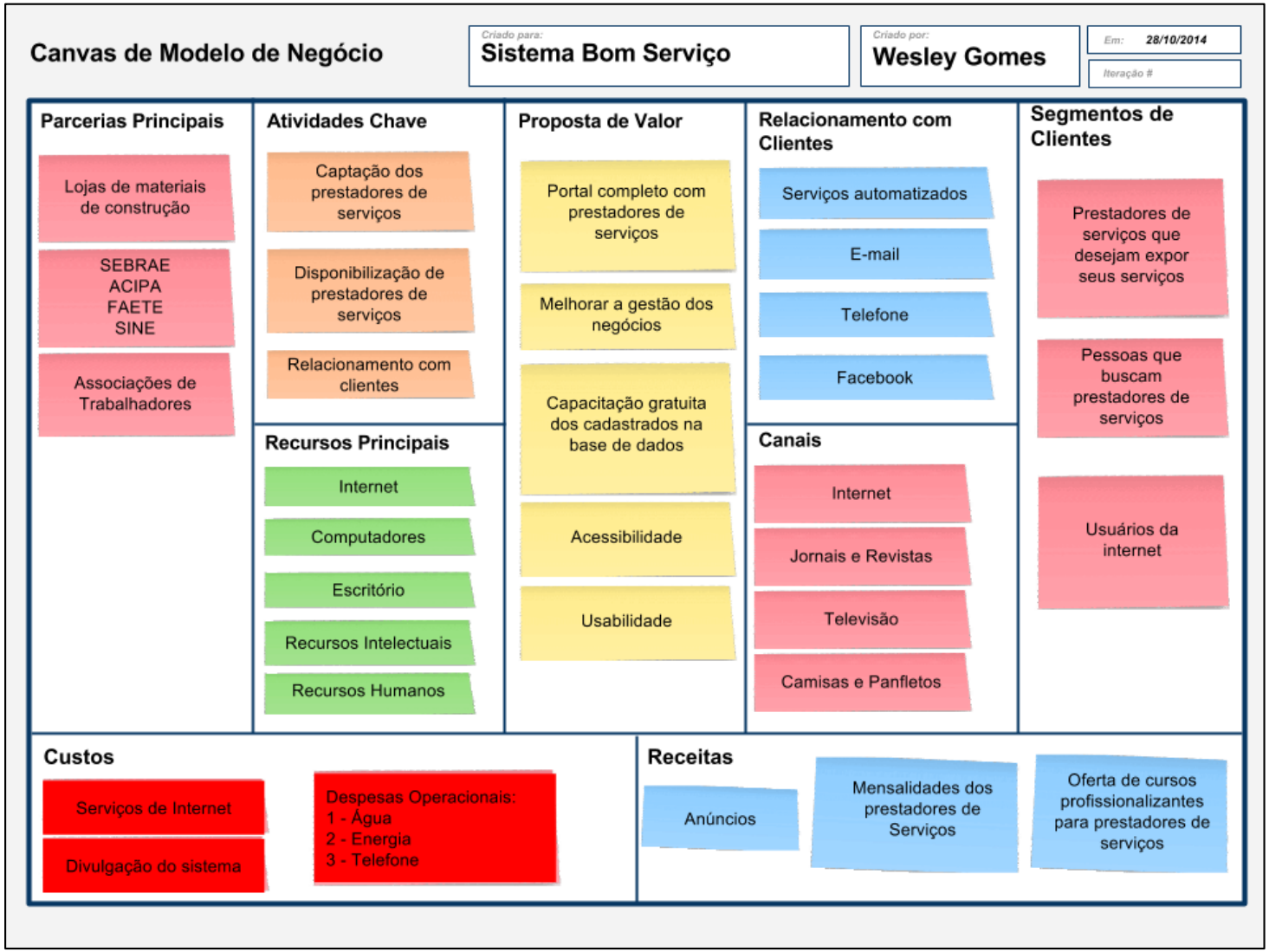


Figura 1. Canvas de Modelo de Negócio

De acordo com a Figura **1**, a intenção é proporcionar uma visão geral do negócio, considerando aspectos relacionados aos seus clientes, oferta, infraestrutura, viabilidade financeira e estabelecer um direcionamento estratégico à gestão do negócio. Com o desenvolvimento do Canvas foi possível identificar os principais pontos chaves do negócio como:

- **Seguimentos de Clientes:** prestadores de serviços autônomos que desejam expor seus serviços e aumentar o alcance de divulgação do mesmo e pessoas que buscam prestadores de serviços;
- **Relacionamento com Cliente:** através de mídias sociais, telefone e e-mails.
- **Proposta de valor:** ter um ambiente integrado que disponibilize de maneira transparente informações de prestadores de serviços a possíveis clientes, melhorar a gestão do negócio através da tecnologia.
- **Atividades chaves do sistema:** relacionamento com clientes, alimentar a base de dados com prestadores de serviços, divulgar prestadores de serviços aos clientes de uma forma rápida e fácil através da internet.
- **Parcerias:** lojas e materiais de construção, onde há um fluxo grande de prestadores de serviços à procura de novos trabalhos, SEBRAE atuando na capacitação dos prestadores de serviços cadastrados na base de dados e associações dos trabalhadores onde muitos trabalhadores tem algum tipo de vínculo.
- **Custos:** telefone, energia, água, serviços de internet e divulgação do sistema.
- **Receitas:** anúncios, mensalidade fixa dos prestadores de serviços, oferta de cursos profissionalizantes.

3 REFERENCIAL TEÓRICO

Nessa seção serão abordados todos os conceitos que fundamentam a criação de serviços utilizando SOA, desde o processo de utilização até o seu funcionamento.

3.1 ARQUITETURA ORIENTADA A SERVIÇO

Ao desvincular o domínio de negócio de tecnologias e modelos específicos, como linguagens ou sistemas operacionais, SOA oferece à organização a chance de acompanhar as mudanças exigidas por seu contexto de negócio sem que isso sobrecarregue o uso dos recursos de Tecnologia da Informação (TI). De acordo com (MARZULLO, 2009) dentro dessa abordagem, podemos identificar benefícios como separação de responsabilidades, organização lógica e facilidade de uso.

Tratando de aplicações baseadas em serviços, SOA é composto de três elementos que representam papéis distintos de interação: o consumidor do serviço, o prestador de serviço e o registro do serviço. Esses três elementos compõem o que chamamos de modelo operacional triangular.

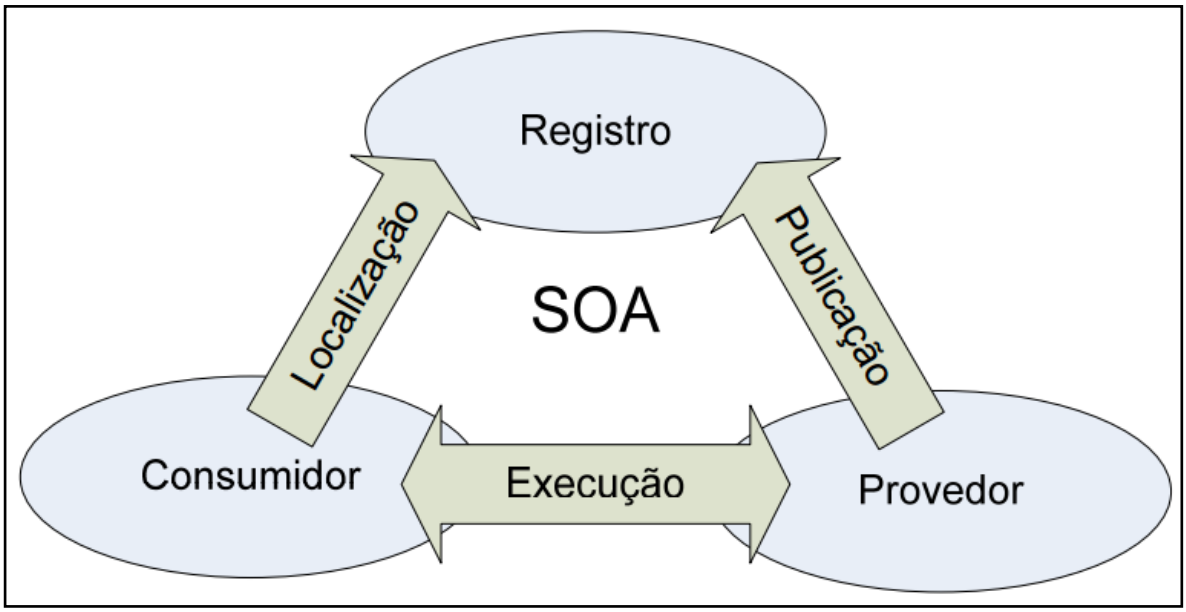


Figura 2**.** Modelo Operacional Triangular SOA (MARZULLO, 2009).

De acordo com Figura **2**, o modelo organizacional de uma arquitetura orientada a serviços se comportam da seguinte forma:

- **Provedor do Serviço:** aquele que oferece o serviço, responsável pela infraestrutura do acesso, determinando todo o seu comportamento.
- **Consumidor do Serviço:** aquele que consome o serviço, responsável pelo comportamento daquele que representa o cliente da organização.
- **Registro do Serviço:** mecanismo que permite ao provedor de serviço cadastrar seus serviços e ao consumidor encontrá-los, normalmente esses registros contêm informações sobre o negócio, informações técnicas com linguagens, tecnologias utilizadas e informações sobre o serviço em si.

De acordo com (KUMAR, 2012), a forma de integrar os dados através dos serviços é um meio de chegar a SOA. Com essa arquitetura, é possível dizer que surgiu uma nova forma de pensar em TI, desaparecendo a ideia de sistemas monolíticos e surgindo a ideia de processos de negócios. Utilizando SOA, os processos podem ser alterados de maneira rápida e eficiente. O tempo de desenvolvimento utilizando em ambientes que passam por problemas de negócios, tais como: gestão de processos, migração de sistemas legados, integração de sistemas e fusão de empresas é encurtado, pois o desenvolvimento das aplicações não é iniciado do zero, aproveitando boa parte do que já existe e fazendo o uso de umas das suas características, que é a reutilização de código.

Segundo (KUMAR, 2012), arquitetura orientada a serviço se torna adequada para a criação de programas em forma de serviços de apoio podendo acontecer à interação através da internet. O conteúdo “serviço” refere-se principalmente às operações envolvendo negócios. Um sistema que tem como proposta utilizar SOA é formado por vários serviços, apresentando geralmente baixo acoplamento por natureza, para caso necessário, novos serviços possam ser adicionados ou os existentes possam ser modificados de maneira rápida, de acordo com a necessidade e dinamismo dos negócios.

Segundo (MARZULLO, 2009), no contexto de SOA, a definição de serviço é:

> “Um serviço é um tipo de relacionamento (contrato) entre um provedor e um consumidor, sendo esse provedor se compromete em realizar determinadas tarefas com resultados pré-estabelecidos para um

> consumidor, e que, por sua vez, se compromete a usar o serviço da forma contratada.”

Antes de concluir com os conceitos, há a necessidade de se definirem mais dois componentes fundamentais para o entendimento da arquitetura SOA: o repositório de serviços e o barramento de serviços. O repositório de serviços é a entidade que provê facilidades para o descobrimento dos serviços disponíveis na arquitetura, sobretudo daqueles fora do escopo temporal e funcional do sistema de informação, ou melhor, do processo de desenvolvimento do nosso sistema. Ele fornece informações como: localização virtual, provedor, taxas, limitações técnicas, aspectos de segurança, entre outras. O barramento de serviços é o meio utilizado para conexão entre todos os participantes da SOA como: serviços e aplicações *frontends*, engloba uma grande diversidade de produtos e conceitos.

De acordo com (MARZULLO, 2009), barramento de serviços deve ser definida de maneira a permitir a integração de aplicações desenvolvidas em diferentes linguagens e plataformas e deve estar alinhada com as atividades comuns em um ciclo de vida SOA.

3.2 WEB SERVICES

De acordo com (MARZULLO, 2009), um *Web Service* representa a materialização da ideia de um serviço que é disponibilizado na internet e que pode ser acessado em qualquer lugar do planeta. Representa uma lógica de negócio que permite que um ou mais clientes enviem requisições de um tipo bem definido de informação e recebam respostas síncronas ou assíncronas.

Os *Web Services* dispõem de soluções viáveis e interessantes se tratando de interoperabilidade entre sistemas totalmente ou parcialmente baseados no modelo cliente-servidor ou até mesmo outros modelos arquiteturais adotados em sistemas distribuídos. Ao usarmos, é possível construir sistemas com reaproveitamento de componentes disseminados no mercado, como banco de dados, serviços comerciais e muito mais (SHARP, 2011).

A demanda por soluções de negócios integradas e distribuídas cresce a cada dia, e a teoria de *web services* vem ao encontro dessas necessidades para dar uma solução

definitiva quanto a forma de se materializar novas estratégias de negócio orientada a serviços. Portanto, o que antes era visto com um repositório de conteúdo, agora se estabeleceu como um repositório de serviços, criando soluções distribuídas e descentralizadas.

Uma definição técnica de *web services* poderia ser como um serviço disponibilizado na internet, descrito via WSDL (Web Services Description Language), registrado via UDDI (Universal Description, Discovery and Integration), acessado utilizando SOAP (Simple Object Access Protocol) e com os dados transmitidos sendo representados em XML (eXtensible Markup Language). A seguir, encontra-se uma breve explicação de algumas tecnologias citadas na definição anterior:

3.2.1 SOAP

Segundo (MARZULLO, 2009) SOAP é um protocolo para troca de informações em ambiente distribuído. É baseado em definições XML. Esse protocolo encapsula as chamadas e retornos aos métodos dos *web services*, sendo utilizado, principalmente, sobre HTTP (Hypertext Transfer Protocol). A Figura **3** ilustra a estrutura do protocolo SOAP.

Figura 3. Estrutura de um Envelope SOAP (MARZULLO, 2009).

Como mostrado na Figura **3**, o elemento envelope é obrigatório, ele define a raiz da mensagem e determina como o documento XML é transportado em uma mensagem SOAP

e como deve ser traduzida pelo *web service* no serviço real. O conceito de *Header* é definido como um cabeçalho opcional. Ele transporta informações adicionais, por exemplo, se a mensagem deve ser processada por um determinado nó intermediário. Quando utilizado, o *Header* deve ser o primeiro elemento do Envelope. Por fim, o *Body* é um elemento obrigatório que irá armazenar o documento a ser transmitido. O elemento *Body* pode conter um elemento opcional Fault, usado para indicar códigos e mensagens de erro que podem ocorrer.

3.2.2 WSDL

Segundo (SAUDATE, 2013), WSDL descreve a interface do serviço de forma estruturada e padronizada usando XML. Ela permite, através da definição de um vocabulário em XML, a possibilidade de descrever serviços e a troca de mensagens. Mais especificamente é responsável por prover as informações necessárias para a invocação do *web service*, como sua localização, operações disponíveis e suas assinaturas. Na Figura **4** é demostrada a estrutura de um WSDL.

Figura 4. Estrutura de um WSDL (ERL, 2011).

- *Types*: Age como um container para definir os tipos de dados usados dentro da mensagem. Permite usar XML *Schema* para definir as estruturas de dados.
- *Message*: permite descrever as mensagens que são trocadas entre o serviço e o consumidor do serviço. Uma mensagem pode possuir várias partes, sendo que cada parte possui um nome e um tipo de dados.
- *PortType*: define o local em que está hospedado o *web service*, identifica o gurpo de ações que podem ser executadas num único ponto de acesso e utiliza o elemento *operation* para representar uma operação dentro desse ponto de acesso.
- *Binding*: define o formato da mensagem e o protocolo de comunicação do elemento *Types*.
- *Service e Ports*: Aparece no final do arquivo WSDL e identifica o serviço que é composto.

3.2.3 UDDI

Um UDDI contém informações categorizadas sobre os serviços e as funcionalidades que eles oferecem, e permite a associação desses serviços com suas informações técnicas, geralmente definidas usando WSDL. Como dito anteriormente, o arquivo de descrição em WSDL descreve as funcionalidades do *web service*, a forma de comunicação e sua localização. A especificação UDDI define uma API (Application Programming Interface) baseada em mensagens SOAP, com uma descrição em WSDL do próprio *web service* do servidor de registro. A maioria dos servidores de registro UDDI também provê uma interface de navegação por browser. De acordo com (CALIENDO, 2010), o UDDI é similar a um motor de busca (como o Google), tem como objetivo ser um mediador do serviço, permitindo que os clientes requisitantes encontrem um fornecedor que melhor atenda suas necessidades.

Desta forma, o UDDI é uma interface web, que determina serviços concedendo a descrição e descoberta de negócios, sendo disponibilizado o acesso e o gerenciamento destes serviços.

3.3 TERCEIRIZAÇÃO DE SERVIÇOS NO TOCANTINS

O Processo de informalidade dos trabalhadores por conta própria pode ser representado como microempresários, desejamos apontar que eles estão criando uma ocupação no mercado de bens, principalmente na prestação de serviços, com o objetivo de se auto-empregar. O que caracteriza esse grupo, especialmente aqueles que operam com baixo nível de produtividade com relação às empresas capitalistas, é que compreende indivíduos com pouco nível de capital físico ou humano, que são simultaneamente patrões e empregados de si mesmos.

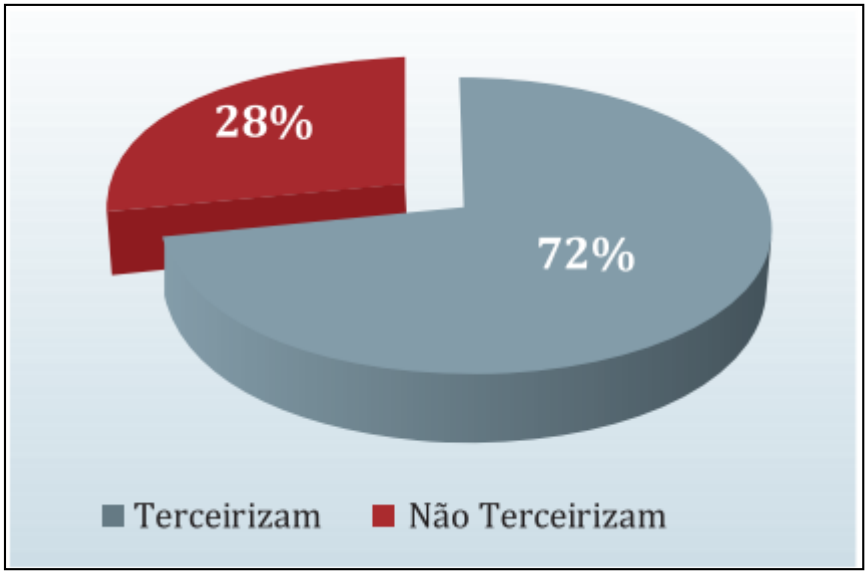


Figura 5. Utilização de Serviços Terceirizados (FIETO).

De acordo com a Figura **5**, é apresentando o resultado de uma pesquisa elabora pela Federação das Indústrias do Estado do Tocantins (FIETO), onde na indústria tocantinense observou-se que um grande percentual de empresas industriais (72%) utiliza e/ou utilizou serviços terceirizados em suas atividades nos últimos três anos. É importante ressaltar que esse é um quadro comum nas empresas industriais brasileiras como um todo. Diante disto, tem-se dado de forma crescente à demanda por contratos com prestadores de serviços terceirizados nos últimos anos, visando à redução de custos e/ou melhoria na qualidade do produto.

Em se tratando da região Norte, se focando no Estado do Tocantins, observa-se que mais da metade das empresas utilizam serviços terceirizados no estado, como já salientado outrora. Neste sentido, de acordo com a FIETO, a pesquisa traz uma abordagem sobre a utilização de serviços terceirizados no Tocantins, baseado na análise da percepção do empresariado tocantinense sobre o desenvolvimento desta variável a contexto estadual, bem como verificar os impactos de fatores como mão de obra, infraestrutura de transporte e fornecimento de energia na produtividade da indústria.

Figura 6. Tipos de Serviços Terceirizado (FIETO).

De acordo com a Figura **6**, em um contexto em nível de estado, observa-se no Tocantins que a grande demanda por serviços terceirizados contratados pelas empresas nos últimos três anos situa-se nos serviços de montagem e/ou manutenção de equipamentos, sendo estes recorrentes em 58% das empresas que terceirizam. É importante estabelecer que a terceirização dos serviços de manutenção no Brasil, mais do que uma tendência é uma realidade em grandes empresas.

3.4 TRABALHOS RELACIONADOS

A lista seguinte descreve sistemas relacionados ao mesmo contexto da aplicação proposta:

- **GetNinjas:** No ar desde maio de 2011, nele e possível encontrar profissionais de diferentes ramos. O sistema lucra com uma porcentagem de cada contrato fechado.
- **Iguanafix:** Acesso a todos os detalhes de profissionais liberais ou empresas. Propostas de orçamentos em menos de 24 horas e possibilidade de comparações de preços.
- **Recomind.net:** O Recomind busca facilitar a contratação de profissionais como faxineiras, pedreiros, eletricistas, mecânicos, médicos e cabeleireiras. A aplicação funciona como um caderno de endereços e telefones com características de rede social.
- **Bougue:** O objetivo é fornecer informações para que os usuários que precisam contratar um serviço possam, por exemplo, comparar propostas. Além das propostas, o usuário pode receber também opiniões de quem já usou os serviços do prestador.
- **ClickARQ:** Um site de concorrência criativa em decoração, arquitetura e design de interiores, ajuda a usuários encontrarem arquitetos. Os clientes solicitam projetos de design de interiores e decoração, e arquitetos e designers enviam suas propostas.

Abaixo dois dos sistemas citados com requisitos mais detalhados, os parâmetros de escolha para uma análise mais aprofundada foram: quantidades de usuários ativos, organização de informações, interface intuitiva, plataformas suportadas e prêmios conquistados.

3.4.1 GetNinjas

GetNinjas aplicação web e mobile, com uma plataforma simples e amigável para que pessoas com habilidade de realizar um serviço específico consigam oferecer seu trabalho para o público em geral e, em contrapartida, para que público possa encontrar de maneira rápida, prática e conveniente os serviços que gostaria, é possível encontrar

profissionais de diferentes ramos, como fotógrafos, babás, assistentes técnicos e diaristas, fornece informações de acordo com sua região, os prestadores de serviço anunciam gratuitamente no portal, que, em compensação, ficam com uma porcentagem de cada contrato fechado. Até a elaboração desse trabalho, segundo Eduardo L'Hotellier, Chief Executive Officer - CEO do GetNinjas, a plataforma reúne mais de 40 mil profissionais cadastrados em 4.000 cidades brasileiras e movimentou cerca R$ 8 milhões em negócios para os profissionais desde seu lançamento, o site recebe uma média de 1.000 orçamentos por dia. Só no Estado de São Paulo, por exemplo, são aproximadamente 10 mil orçamentos por mês. Conquistou prêmios como Startup do ano de 2012 promovido pela TheNextWeb Startup e o reconhecimento do Jornal americano The New Work Times em 2011. Sua interface gráfica está observada na Figura **7.**

Figura 7. Interface web do GetNinjas (GETNINJAS - 2011).

3.4.2 Recomind.net

Recomind.net aplicação web e mobile, apresenta produtos e serviços baseado nas indicações da rede de contatos do usuário, conta com a integração com o facebook, através dele o usuário tem um espaço para preencher as informações sobre o profissional ou serviço que procura. A partir disso, a requisição ficará listada para que outros usuários possam respondê-la. Além disso, o usuário terá a opção de postar as informações no mural de sua rede social, para que seus amigos possam indicar um prestador de serviço que se adeque as suas necessidades.

Um serviço Multiplataforma roda em tablets, smartphones Android, Iphone e computadores. Um dos vencedores da edição 2011 do Desafio Buscapé: Sua Ideia Vale Um Milhão. O concurso premiou a startup com um investimento de R$ 300 mil, atualmente a aplicação pertence ao grupo Buscapé company. A Figura **8** apresenta sua interface gráfica.

Figura 8. Interface web do Recomind.net (RECOMIND.NET - 2011).

4 METODOLOGIA

A metodologia utilizada para a realização deste trabalho está dividida nas seguintes etapas:

- Pesquisa e coleta de dados;
- Modelagem;
- Desenvolvimento;
- Documentação;

Na etapa de Pesquisa e coleta de dados, foi realizada uma revisão bibliográfica baseada na leitura de textos disponíveis em livros e artigos técnico-científicos e criação do modelo de negócio utilizando Bussines Model Canvas, realizou então um questionário destinado a coleta de dados sobre os prestadores de serviço e clientes, por meio da ferramenta Google Docs[1] que, além de permitir a criação de documentos, planilhas, apresentações, e desenhos, oferece o recurso de criar formulários online, onde um ou mais usuário que acessam o endereço virtual do formulário podem preencher e enviar suas respostas que estão disponíveis ao proprietário e colaboradores quando houver. Tais resultados serviram para validação dos requisitos levantados até o momento e, possivelmente, a descoberta de outros, o resultado da pesquisa é apresentado em forma de gráficos no apêndice A deste trabalho, a etapa de pesquisa busca organizar os dados de forma que se possa comparar os sistemas de mesmo contexto e extrair características interessantes de cada um.

A etapa de Modelagem descreve os principais diagramas UML (Unified Modeling Language) para caracterização formal do sistema proposto. Segundo (BOOCH, G. and RUMBAUGH, J. and JACOBSON, 2006), UML é uma linguagem gráfica para visualização, especificação, construção e documentação de artefatos de sistemas de software. A UML proporciona uma forma padrão para a preparação de planos de arquitetura de projetos de sistemas, os diagramas ilustram graficamente a arquitetura, estrutura, interações e comportamento do sistema proporcionando uma abstração de vários contextos inerentes ao software. Para a explicação do banco de dados, foi criado um dicionário de dados com o intuito de detalhar minuciosamente cada campo de cada tabela do banco de dados.

Na etapa de Desenvolvimento foram realizadas todas as configurações de ambiente necessárias, desenvolvimento do *Web Service*, criação dos serviços e aplicação de teste para consumo dos serviços desenvolvidos.

A etapa de Documentação foi desenvolvida de forma paralela tanto na pesquisa, quanto na modelagem e desenvolvimento, durante a organização de dados, tabulação de dados, confecção dos diagramas que modela o sistema.

Todos os experimentos foram elaborados em locais de apoio como o Núcleo de Tecnologia da Informação (NTI), ligado ao curso de Sistemas de Informação da Faculdade Católica do Tocantins (FACTO). O tempo para realização do projeto teve cronograma próprio definido no pré-projeto da pesquisa. Os materiais essenciais para a realização deste trabalho se baseiam em ferramentas gratuitas e versões comunitárias. Utilizou-se a Integrated Development Environment (IDE) ECLIPSE LUNA para desenvolvimento do *Web Service*, para a criação do sistema web para consumo dos serviços foi utilizado a IDE NetBeans 8.0.1. O banco de dados adotado para o ambiente de produção do sistema foi o MySQL. Foi utilizado o Sistema de Gerenciamento de Banco de Dados (SGDB) MySQL Workbench 6.0. A ferramenta Workbench possui as vantagens de ser gratuita, e ser compatível com todos os sistemas operacionais, possui uma documentação bem detalhada e com uma linguagem simples, permite fazer a engenharia reversa caso necessário e exporta os dados em vários formatos, como por exemplo, PNG e PDF. Nesse projeto, a ferramenta MySQL Worbench foi utilizada para a criação das tabelas e seus respectivos dados, e também para a sincronização do banco de dados. Por fim, na criação do *Web Service* foi utilizado o Apache Axis e configurado o TOMCAT 7.0 como servidor web.

[1]Google Docs. Disponível em https://docs.google.com/

5 DESENVOLVIMENTO

Este capítulo apresenta a implementação do sistema proposto relacionando os conceitos e teorias que o fundamentam. Métodos e ferramentas que foram utilizadas no desenvolvimento da proposta são brevemente descritos.

5.1 VISÃO GERAL DA FERRAMENTA

Os serviços desenvolvidos durante este trabalho é uma proposta para melhorar a acessibilidade de informações sobre prestadores de serviços autônomos no Tocantins.

Todos os dias, milhares de pessoas precisam de profissionais como: pintores, eletricistas, domesticas, jardineiros entre outros, como encontrar esse tipo de profissional de forma rápida fácil e confiável? Esta realidade está presente na maioria das cidades brasileiras. Neste cenário a cidade Palmas, capital do estado do Tocantins não deixa de estar incluída, pois é notória tal situação no cotidiano de boa parte de seus cidadãos.

Quando um usuário necessita de informações concernentes a prestadores de serviços qualquer pessoa é consultada a fim de extrair alguma informação útil para sua necessidade o que pode não lhe informar dados precisos e de confiança. Quando alguém não é abordado para fins de consulta de informações, o usuário só poderá adquirir informações em órgão público como o Sistema Nacional de Emprego - SINE. Com essa realidade apresentada, não há nada mais conveniente do que obter informações de prestadores de serviços, qualificações de serviços, localização de prestadores entre outros, sem importar o lugar e o momento. O sistema proposto visa à disponibilização de informações dessa natureza, além de recursos que auxiliarão qualquer pessoa a localizar um prestador de serviço com mais facilidade e garantia.

5.2 REQUISITOS FUNCIONAIS

Segundo Somerville (2011, p. 59), requisitos funcionais são declarações de funções que o sistema deve fornecer, de como o sistema deve reagir a entradas específicas e como deve se comportar em determinadas situações. Com isso, abaixo uma tabela contendo todos os requisitos funcionais dos sistemas relacionados e do sistema proposto.

Tabela 1. Requisitos Funcionais

ID	Requisitos	Bom Serviço	GetNinjas	Iguanafix	Recomind.net	Bougue	ClickARQ
RFC0001	Cadastrar Usuário	X	X	X	X	X	X
RFC0002	Cadastrar Serviço	X	X	X	X	X	X
RFC0003	Cadastrar Empresa	-	-	X	-	-	-
RFC0004	Buscar Serviço	X	X	X	X	X	X
RFC0005	Alterar Perfil de Usuário	X	X	X	X	X	X
RFC0006	Alterar Perfil de Serviço Prestado	X	X	X	X	X	X
RFC0007	Remover Usuário	X	X	X	X	X	X
RFC0008	Remover Serviço	X	X	X	X	X	X
RFC0009	Localização do Prestador de Serviço mais próximo.	X	-	-	X	-	-
RFC0010	Avaliação do Serviço Prestado	X	X	X	X	X	X
RFC0011	Suporte a Dúvidas	X	X	X	X	X	X
RFC0012	Autenticação com Facebook	X	X	X	X	X	-
RFC0013	Gamificação de Avaliação	X	X	X	X	X	-
RFC0014	Indicação do Sistema	X	-	-	-	-	-

Para um melhor entendimento da Tabela **1**, é feito uma descrição mais detalhada dos requisitos funcionais:

- **RFC0001 - Cadastrar Usuário**: Requisito responsável por cadastrar todos os usuários que solicitem e/ou que pretendem ofertar algum tipo de serviço, essa condição é utilizada em todos os sistemas abordados.
- **RFC0002 - Cadastrar Serviço**: Requisito responsável por cadastrar todos os serviços prestados no sistema, este quesito é adotado por todos os sistemas levantados, entretanto para o sistema Iguanafix e necessário primeiro a execução do requisito RFC0003.
- **RFC0003 - Cadastrar Empresa**: Requisito responsável por cadastrar uma empresa prestadora de serviços, essa condição é utilizada somente no sistema Iguanafix pelo fato do sistema fazer uma avaliação do prestador de serviço antes de sua liberação, com isso a empresa prestadora de serviço garante algumas vantagens como: ter uma lista segura de profissionais certificados, gerando mais confiança aos clientes.
- **RFC0004 - Buscar Serviço**: Requisito responsável por buscar qualquer tipo de serviço no sistema como: pesquisar um serviço de eletricista, um pedreiro, uma diarista, este item é utilizado por todos os sistemas abordados.
- **RFC0005 - Alterar Perfil de Usuário**: Requisito responsável por fazer alterações no perfil de usuários como: alterar nome, trocar foto, alterar endereço, este quesito é utilizado por todos os sistemas levantados.
- **RFC0006 - Alterar Perfil de Serviço Prestado**: Requisito responsável por fazer alterações no serviço ofertado como: alterar o nome do serviço, alterar o valor do serviço, este quesito é adotado por todos os sistemas levantados, entretanto para o sistema Iguanafix e necessário primeiro a execução do requisito RFC0003.
- **RFC0007 - Remover Usuário**: Requisito responsável por remover qualquer usuário devidamente cadastrado no sistema, onde o próprio usuário poderá excluir sua conta, excluindo todos os dados do mesmo, está condição é aplicada por todos os sistemas abordados.
- **RFC0008 - Remover Serviço**: Requisito responsável por remover qualquer serviço devidamente cadastrado no sistema, o próprio prestador de serviço poderá excluir o serviço, este quesito é adotado por todos os sistemas levantados.

- **RFC0009 - Localização do Prestador de Serviço mais próximo**: Requisito responsável por exibir a localização exata do prestador de serviço mais próximo do usuário que efetuou a pesquisa, está condição e aplicada apenas no sistema Recomind.net, o desenvolvimento desse requisito faz parte do diferencial do projeto proposto, devido ao grau de detalhamento que exibirá as informações.
- **RFC0010 - Avaliação do Serviço Prestado**: Requisito responsável por avaliar o serviço prestado através de comentários, onde após o serviço pronto, o usuário solicitante poderá fazer algum comentário a respeito do mesmo, este quesito é abordado por todos os sistemas levantados.
- **RFC0011 - Suporte a Dúvidas**: Requisito responsável por dá suporte a todos os usuários, através de e-mails e/ou telefone, esse item é utilizado por todos os sistemas abordados.
- **RFC0012 - Autenticação com Facebook**: Requisito responsável por agilizar o processo de autenticação de usuários através do facebook, é compartilhar informações através da rede social, quesito que apenas o sistema ClickARQ não utiliza, pelo fato que os usuários deverão fornecer informações pessoais como nome, endereço, número de telefone, bem como outros dados pessoais relativos aos meios de pagamento utilizados, como: número de cartão de crédito, todas as informações pessoais fornecidas voluntariamente ou através de procedimentos automatizados estão sujeitas a medidas de segurança para impedir acesso, uso e divulgação não autorizados, estão sujeitas às disposições presentes na Política de Privacidade do Site.
- **RFC0013 - Gamificação de Avaliação:** Requisito responsável por gamificar à forma de avaliar os prestadores de serviços como: avaliações de classificação com estrelas para os melhores serviços prestados, quesito que apenas o sistema ClickARQ não utiliza, fato de não ter uma funcionalidade diferente do requisito RFC0010, o desenvolvimento desse requisito faz parte do diferencial do projeto proposto onde os prestadores que obtiverem mais avaliações positivas desse tipo terá vantagens especiais como: exibição na página principal da aplicação por mais tempo, descontos exclusivos na utilização do sistema.
- **RFC0014 - Indicação do Sistema:** Requisito responsável pela indicação da ferramenta, permite que o usuário possa indicar através de um e-mail o sistema para outras pessoas aumentando assim o poder de divulgação do sistema, está condição e aplicada apenas no sistema proposto, o desenvolvimento desse requisito faz parte

do diferencial do projeto, devido ao grau de detalhamento que exibirá as informações.

5.3 REQUISITOS NÃO-FUNCIONAIS

Além dos requisitos funcionais existem aqueles que não têm relação direta com as funções do software. Suas relações podem estar ligadas a propriedades da organização, segurança, desempenho, facilidades de uso e outras. Esses são os requisitos não-funcionais. Tais requisitos também são definidos por Somerville (2011, p. 59), ao qual diz que são restrições sobre os serviços ou as funções oferecidas pelo sistema. Entre eles destacam-se restrições de tempo, restrições sobre o processo de desenvolvimento, padrões, entre outros. Para tal, foram identificados requisitos não-funcionais, logo abaixo tabelas representando todos os requisitos dos softwares analisados e da arquitetura proposta.

Tabela 2. Requisitos não-funcionais de Segurança.

ID	Requisitos	Software Proposto	GetNinjas	Iguanafix	Recomind.net	Bougue	Click ARQ
RNF/SEG-01	O sistema deverá restringir o acesso às informações pessoais dos usuários é prestadores de serviços cadastrados.	X	X	X	X	X	X
RNF/SEG-02	Utilizar técnicas de backup de dados.	X	X	X	X	X	X
RNF/SEG-03	Utilizar técnicas de restauração de dados em caso de falhas.	X	X	X	X	X	X
RNF/SEG-04	Utilizar mecanismos de proteção contra escrita, que previnam alterações indevidas e mantenham a integridade dos dados armazenados.	X	X	X	X	X	X

Para um melhor entendimento da Tabela **2**, é feito uma descrição detalhada dos requisitos de segurança:

- **RNF/SEG-01**: Requisito responsável por restringir o acesso a pessoas não autorizadas a informações pessoais de usuários como: nome, CPF, RG, endereço, telefone, etc., através de privilégios e métodos para o bloqueio ao sistema, esse item e utilizado por todos os sistemas abordados.
- **RNF/SEG-02:** Requisito responsável por executar o backup semanal de todos os dados do sistema em algum servidor diferente do que hospeda a aplicação, de preferência em um local externo a organização, esse quesito é abordado por todos os sistemas levantados.
- **RNF/SEG-03**: Requisito responsável por garantir que os dados sejam restaurados em casos de falhas técnicas é roubos de servidores, essa condição é utilizada por todos os sistemas abordados.
- **RNF/SEG-04**: Requisito responsável por proteger os dados de alterações indevidas por pessoas não autorizadas, esse quesito é utilizado por todas as aplicações abordadas.

Tabela 3. Requisitos não-funcionais de Performance.

ID	Requisitos	Software Proposto	GetNinjas	Iguanafix	Recomind.net	Bougue	Click ARQ
RNF/PER-05	O banco de dados deve ser atualizado em tempo real.	X	X	X	X	X	X
RNF/PER-06	O tempo de retorno das consultas (isto é, o intervalo de tempo entre qualquer consulta e seu resultado) não pode ser maior do que 7 segundos.	X	-	-	X	-	-
RNF/PER-07	O sistema deve permitir acesso de qualquer local que disponha de um computador com acesso à Internet.	X	X	X	X	X	X

Para um melhor entendimento da Tabela **3**, é feito uma descrição detalhada dos requisitos de performance:

- **RNF/PER-05**: Requisito responsável pela atualização em tempo real dos dados cadastrado no banco de dados, esse quesito é utilizado por todos os softwares abordados.
- **RNF/PER-06**: Requisito responsável por exibir o resultado de uma busca com o tempo inferior a 7 segundos, evitando assim transtorno para usuários que aguardam por um resultado, está condição e aplicada apenas no sistema Recomind.net, o desenvolvimento desse requisito faz parte do diferencial da aplicação proposta, devido a forma de implementação do banco de dados, garantindo rapidez em consultas complexas.
- **RNF/PER-07**: Por se tratar de uma aplicação totalmente web, este requisito é responsável pelo acesso ao sistema de qualquer computador com acesso a internet, essa condição é utilizada em todas as aplicações abordadas.

Tabela 4. Requisitos não-funcionais de Usabilidade.

ID	Requisitos	Software Proposto	GetNinjas	Iguanafix	Recomind.net	Bougue	Click ARQ
RNF/USA-08	O sistema será operado em ambiente totalmente web.	X	X	X	X	X	X
RNF/USA-09	O sistema usará uma interface intuitiva que permitirá a utilização do sistema com toda sua potencialidade em um curto espaço de tempo e beneficiará o trabalho dos usuários.	X	X	X	X	X	-
RNF/USA-10	As mensagens de erros devem ser claras e concisas para o entendimento perfeito do usuário do sistema.	X	X	X	X	X	X

Para um melhor entendimento da Tabela **4**, é feito uma descrição detalhada dos requisitos de usabilidade:

- **RNF/USA-08**: Requisito garante que a aplicação seja operada somente em ambiente web, essa condição é utilizada em todos os sistemas levantados.

- **RNF/USA-09**: Requisito responsável pela utilização de interfaces intuitiva, garantindo que o usuário entenda todos os passos para uma boa utilização do sistema, potencializando todas a vantagens que beneficia os usuários, esse item é utilizado por todos os sistemas abordados exceto o ClickARQ, análise feita na usabilidade dessa aplicação constatou uma má usabilidade das interfaces para o processo de sua utilização.
- **RNF/USA-10**: Requisito responsável por exibir de forma clara para os usuários, mensagens de erros como: preenchimento indevido de campos, essa condição é utilizada em todos os sistemas abordados.

Tabela 5. Requisitos não-funcionais de Manutenabilidade.

ID	Requisitos	Software Proposto	GetNinjas	Iguanafix	Recomind.net	Bougue	Click ARQ
RNF/MAN-11	O sistema será dividido em módulos, de modo que as atualizações serão feitas mais rapidamente e sem a necessidade de longos períodos de atualização onde o sistema ficaria impossibilitado de operar.	X	X	X	X	X	X

Para um melhor entendimento da Tabela **5**, é feito uma descrição detalhada do requisito de manutenabilidade:

- **RNF/MAN-11**: Requisito responsável por modularizar à aplicação, de modo que as atualizações futuras possam ser feitas apenas naquele modulo que a necessite, evitando que a aplicação fique longo período impossibilitado de operar, essa condição é utilizada em todos os sistemas levantados.

Tabela 6. Requisitos não-funcionais de Documentação.

ID	Requisitos	Software Proposto	GetNinjas	Iguanafix	Recomind.net	Bougue	Click ARQ
RNF/DOC-12	O sistema possuir um manual de uso a fim de auxiliar os diferentes tipos de usuário. O mesmo explicará	X	X	X	X	X	X

	detalhadamente como proceder na realização das funções requisitadas para a aplicação.						
RNF/DOC-13	O sistema possuir vídeos demonstrativos do funcionamento da ferramenta.	X	X	-	-	-	X

Para um melhor entendimento da Tabela **6**, é feito uma descrição detalhada dos requisitos de documentação:

- **RNF/DOC-12**: Requisito responsável por disponibilizar um documento do software com todas as informações possíveis como: tutoriais de cadastros e consultas, preços de utilização, informações de contato, etc., de forma a maximizar o uso da ferramenta web, essa condição é utilizada em todos os sistemas levantados.
- **RNF/DOC-13**: Requisito responsável por exibir vídeos informativos da ferramenta, para usuários leigos, buscando auxiliar na utilização da ferramenta, a análise de usabilidade constatou que essa condição é utilizada apenas nos sistemas: GetNinjas e ClickARQ, o desenvolvimento desse requisito faz parte do diferencial do projeto proposto, devido a uma forma diferente de demonstrar o funcionamento da aplicação através de vídeos interativos.

5.4 MODELAGEM

Afim de demonstrar graficamente a estrutura dos requisitos do software, nessa seção serão abordados diagrama de componentes, atividade, sequência e diagrama do banco de dados, por conter diversos diagramas de caso de uso ele é apresentado no apêndice A deste trabalho juntamente com os casos de uso expandidos.

5.4.1 Diagrama de Atividade

De acordo com (GUEDES, 2011), diagrama de atividade preocupa-se em descrever os passos a serem percorridos para a conclusão de uma atividade específica, podendo esta ser representada por um método com certo grau de complexidade, um algoritmo, ou mesmo por um processo completo. O diagrama de atividade concentra-se na representação do fluxo de controle de uma atividade.

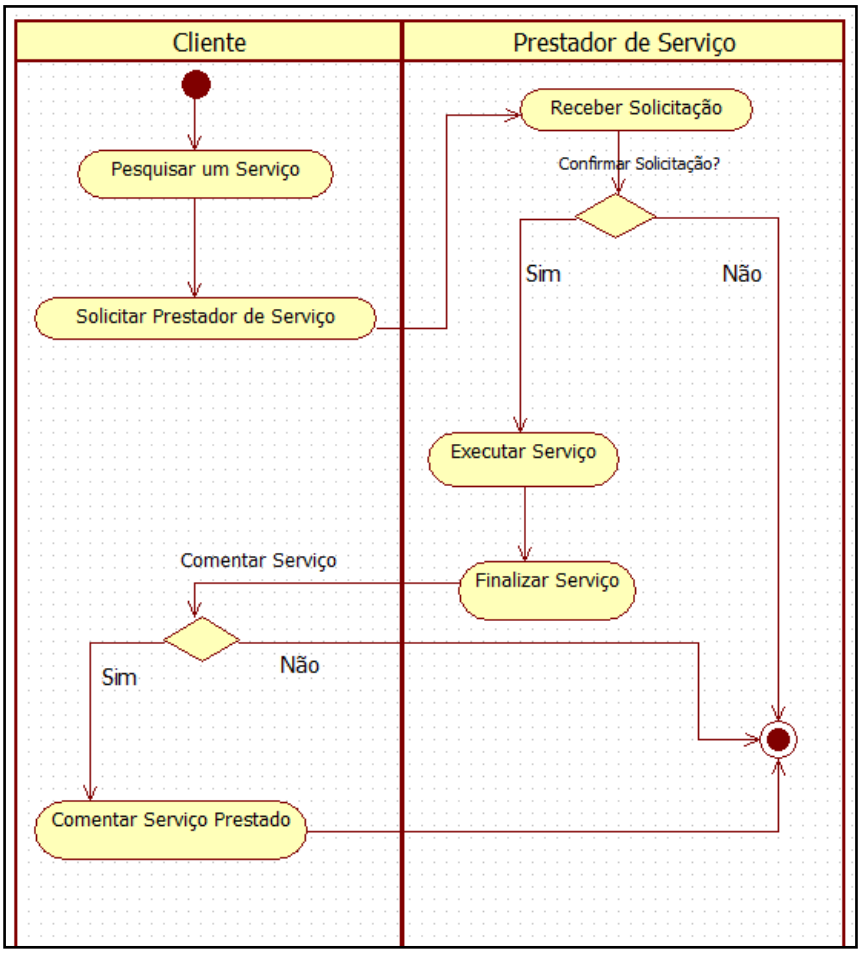


Figura 9. Diagrama de Atividades.

A Figura **9** apresenta o diagrama necessário para contratação de um serviço com os seguintes passos:

1. O cliente pesquisa um determinado serviço;
2. Com o resultado da pesquisa e possível visualizar perfis dos prestadores de serviços com comentários de serviços já prestados, assim poderá solicitar um melhor prestador de serviço;
3. O prestador recebe a solicitação e decide se deseja aceitar ou não;
4. Caso não aceite o processo e finalizado;
5. Caso aceite executa o serviço;
6. Após o serviço concluído notifica no sistema a conclusão do mesmo;
7. O cliente recebe uma notificação para comentar o serviço prestado;
8. Caso não aceite o processo e finalizado;
9. Caso aceite é feito um comentário sobre a qualidade do serviço prestado;

10. Finaliza o processo da aplicação;

5.4.2 Diagrama de Componentes

Figura **10** apresenta os componentes do sistema em módulos de código-fonte, formulários, módulos executáveis etc. e determina como tais componentes estarão estruturados e irão interagir para que o sistema funcione de maneira adequada.

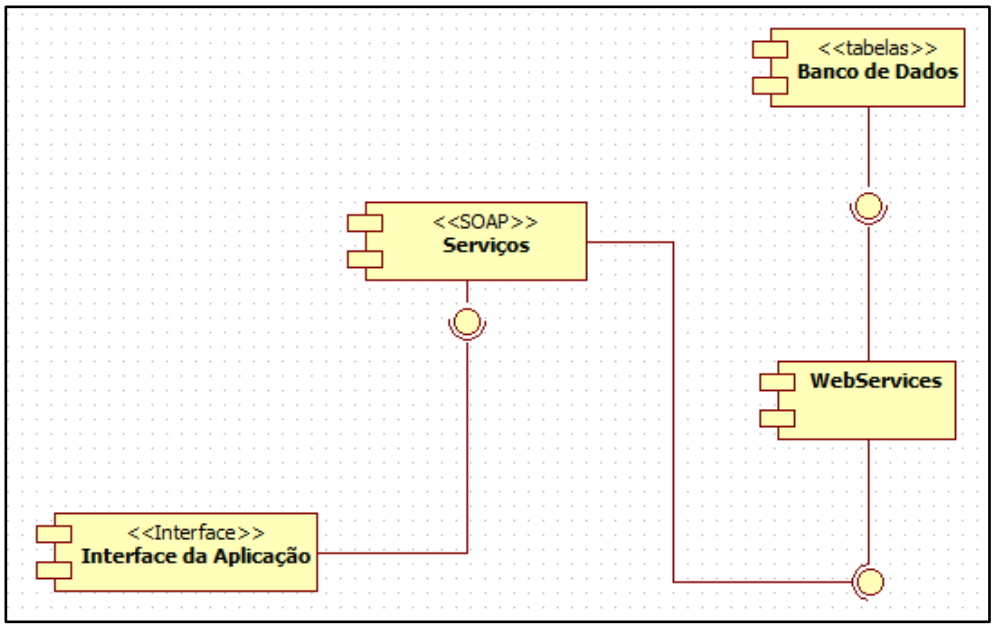


Figura 10. Diagrama de Componentes.

A Figura **10** descreve o diagrama da seguinte forma:

- O componente banco de dados é implementado de acordo com os requisitos levantados na seção 5.2 deste trabalho. É responsável por persistir e recuperar os dados;
- Através do componente *Web Services* é feito a comunicação entre o banco de dados e os serviços implementados;
- O componente Serviços é responsável pelo desenvolvimento de todos os serviços necessários para a utilização através de outras interfaces;
- Componente Interface da Aplicação é o executável que requer uma interface de acesso, este processo de padronização tem em vista a exibição de páginas Web em diversos dispositivos (computadores, smartphone, televisão, tablets, etc);

5.4.3 Diagrama de Sequência

De acordo com (GUEDES, 2011), diagrama de sequência é um diagrama comportamental que preocupa-se com a ordem temporal em que as mensagens são trocadas entre os objetos envolvidos em um determinado processo. Em geral, baseia-se em um caso de uso de diagrama de classes para determinar os objetos das classes envolvidas em um processo.

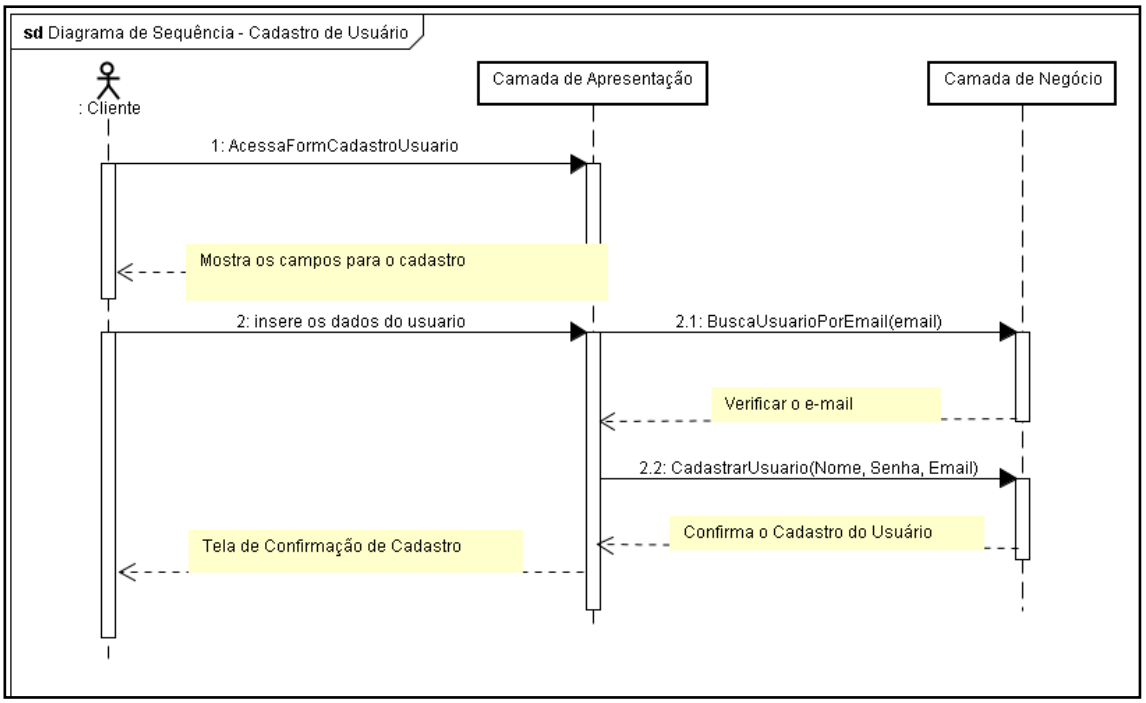


Figura 11. Diagrama de Sequência de Cadastro de Usuário.

A Figura **11** exibe o fluxo do diagrama de sequência de cadastro de cliente para um entendimento técnico, onde o usuário acessa o formulário de cadastro, a aplicação exibe os campos para cadastro, o usuário insere os dados requeridos, a camada de negócio busca por um e-mail para verificar se já existe cadastrado, após a verificação efetua o cadastro é por fim exibe uma mensagem de confirmação para o usuário.

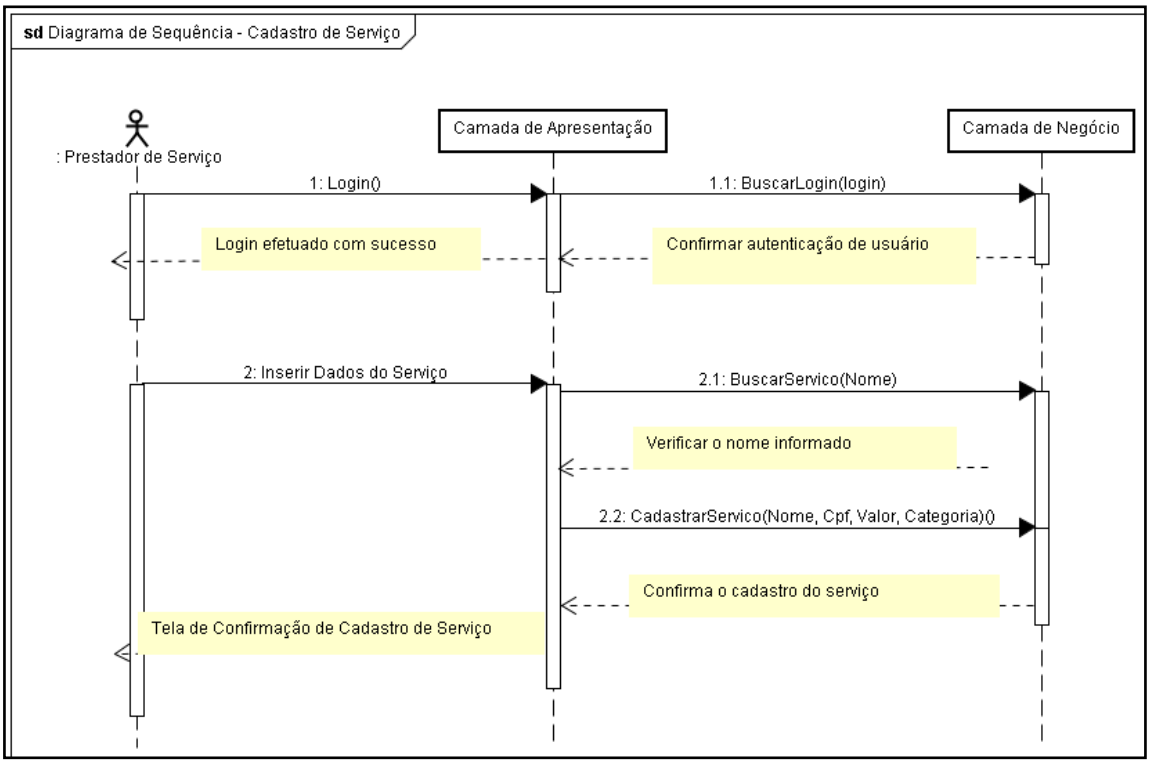


Figura 12. Diagrama de Sequência de Cadastro de Serviço.

A Figura **12** exibe o fluxo do diagrama de sequência de cadastro de serviço para um entendimento técnico, onde o usuário efetua o login, a camada de apresentação busca por um login válido, autentica o usuário e efetua o login, a aplicação exibe os campos para cadastro de serviço, o usuário insere os dados requeridos, a camada de negócio busca por um serviço para verificar se já existe cadastrado, após a verificação efetua o cadastro é por fim exibe uma mensagem de confirmação de cadastro de serviço para o prestador.

5.4.4 Diagrama do Banco de Dados

De acordo com (RAMEZ e NAVATHE, 2005) as vantagens na utilização de um padrão para a modelagem de dados é extremamente necessária. A criação de um modelo único faz-se necessário seguindo os seguintes critérios:

- Tabelas serão nomeadas sempre no singular, sem acentos e no lugar do espaço será usado o caractere underline (_) entre os nomes das mesmas, sendo que as letras serão minúsculas;

- Todas as tabelas terão uma chave primária com auto incremento para identificar um registro específico;
- Os relacionamentos serão estabelecidos com chaves estrangeiras, que serão nomeadas com o prefixo “fk” seguido de “_” e o nome da chave primária da outra tabela concatenado com nome da tabela relacionada;

A Figura **13** demonstra como as tabelas estão relacionadas umas com as outras:

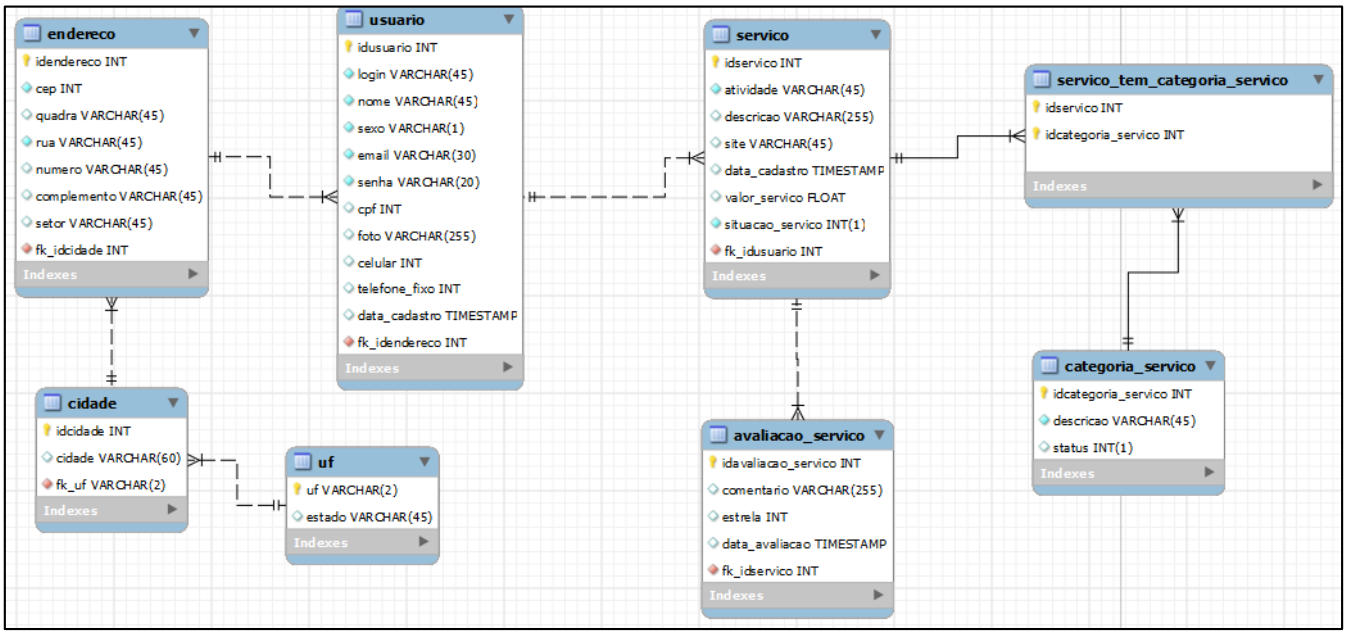


Figura 13. Diagrama do Banco de Dados.

Como mencionado na metodologia, para uma melhor compreensão humana sobre como os dados estão relacionados foi criado um dicionário de dados que é apresentado apêndice C deste trabalho, detalhando cada tabela do banco de dados.

5.5 CODIFICAÇÃO DO SISTEMA

Nessa seção serão apresentados os elementos do *Web Service* desenvolvido, juntamente com os serviços criados, o sistema web de teste para o consumo dos serviços como especificado nos objetivos que se encontra na seção 1.3.1, bem como as ferramentas utilizadas e os principais trechos de código.

5.5.1 Criação dos Serviços

Para iniciar o desenvolvimento dos serviços, foi necessário instalar as ferramentas e o ambiente de desenvolvimento, estes já apresentados na seção 4 em que introduz sobre quais ferramentas foram utilizadas e quais os fatores determinantes para a escolha dessas ferramentas na realização do trabalho proposto.

De maneira a se pensar sobre a organização da codificação dos serviços criados neste projeto, foram gerados alguns pacotes como ilustra a Figura **14**.

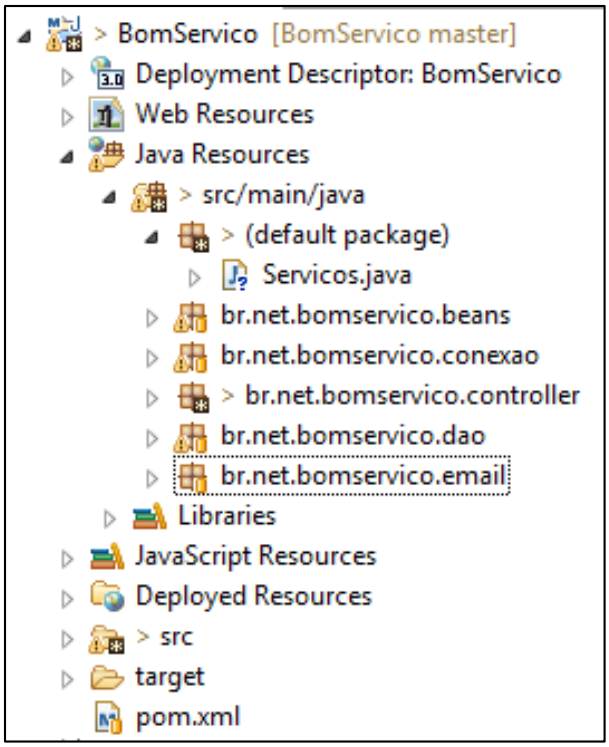


Figura 14. Estrutura do projeto para criação dos serviços.

Como pode ser notado na figura acima, o projeto segue o padrão MVC (Model View Controller) exceto pela View que não é necessária. A classe *Servicos* é responsável pela criação dos serviços que serão consumidos pelos clientes, o pacote *br.net.bomservico.beans* reflete as tabelas do banco de dados e as operações sobre essas tabelas, o pacote *br.net.bomservico.conexao* é responsável pela comunicação da aplicação com o banco de dados, o pacote *br.net.bomservico.controller* representa o Controller para as exposições dos serviços do *Web Service*, onde se encontra toda a regra de negócio da aplicação, o pacote *br.net.bomservico.dao* é designado para exercer a função de manipular

os dados desse banco e o pacote *br.net.bomserivco.email* refere-se a configuração de envio de e-mails utilizado pelo sistema.

Na Figura **15** é possível observar com mais detalhes o desenvolvimento de alguns serviços como *CadastrarUsuario*, *CadastrarServico* e *PesquisarServico*.

```
13      // Cadastrar o Usuario
14      public void CadastrarUsuario(String nome, String sobrenome, String login,
15              String senha) {
16          Cadastro cadastro = new Cadastro();
17
18          try {
19              cadastro.cadastroCliente(nome, sobrenome, login, senha);
20          } catch (IOException e) {
21              e.printStackTrace();
22          }
23      }
24
25      // Cadastrar o Servico
26      public void CadastrarServico(String atividade, String descricao,
27              String site, Calendar data_cadastro, float valor, int celular,
28              int telefone_fixo) {
29
30          Cadastro_Servicos cadastro_Servicos = new Cadastro_Servicos();
31          try {
32              cadastro_Servicos.cadastroServico(atividade, descricao, site,
33                      data_cadastro, valor, celular, telefone_fixo);
34          } catch (Exception e) {
35              e.printStackTrace();
36          }
37      }
38
39      // Pesquisar o Servico
40      public void PesquisarServico(String atividade, String cidade) {
41
42          Pesquisar pesquisar = new Pesquisar();
43          try {
44              pesquisar.pesquisar(atividade, cidade);
45          } catch (Exception e) {
46              e.printStackTrace();
47          }
48      }
```

Figura 15. Classe responsável pela criação dos serviços.

Como é possível verificar, na linha 14 da Figura **15**, o método *CadastrarUsuario* possui parâmetros que devem ser passados para que possa ocorrer o cadastro de maneira correta, a sua execução. Nesse método como em todos os outros, existe uma variável que instancia a classe responsável pela comunicação com o banco de dados, representada na

linha 16, possibilitando que o cliente não se preocupe de como é feita a comunicação com o mesmo, como foi dito no início dessa seção.

Para resolver problemas como gerenciamento de dependências, controle de versão de artefatos, gerar relatórios de produtividade foi utilizado uma ferramenta de gerenciamento de projetos de software chamada Apache Maven, para configurar um projeto com o Maven, foi utilizado um arquivo chamado *pom.xml*. Este tem formato XML e deve ser criado na pasta raiz do projeto. Toda configuração é feita neste arquivo e todas as fases da construção são realizadas com a execução de plug-ins, que serão empregados para testar seu projeto, compilá-lo, empacotá-lo, integrá-lo a outras ferramentas, dentre diversas outras funcionalidades.

5.5.2 Criação do Web Service

Define-se Web Service como um serviço que é disponibilizado na internet, especificado pelos protocolos WSDL, registrado via UDDI, acessado utilizando SOAP e com os dados transmitidos sendo representados em XML como descrito na seção 3.2.

A criação do *Web Service* neste projeto foi realizada através Apache Axis, com ele é possível transformar qualquer classe java um serviço web, ele é implementado como um componente web que processa todas as requisições SOAP que cheguem ao contêiner.

```
▼<wsdl:definitions xmlns:apachesoap="http://xml.apache.org/xml-soap" xmlns:impl="http://localhost:8081/axis/Servicos.jws"
 xmlns:intf="http://localhost:8081/axis/Servicos.jws" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="http://localhost:8081/axis/Servicos.jws">
  ▼<!--
     WSDL created by Apache Axis version: 1.4
     Built on Apr 22, 2006 (06:55:48 PDT)
   -->
   <wsdl:message name="LogarResponse"></wsdl:message>
  ▼<wsdl:message name="CadastrarServicoRequest">
     <wsdl:part name="atividade" type="xsd:string"/>
     <wsdl:part name="descricao" type="xsd:string"/>
     <wsdl:part name="site" type="xsd:string"/>
     <wsdl:part name="data_cadastro" type="xsd:dateTime"/>
     <wsdl:part name="valor" type="xsd:float"/>
     <wsdl:part name="celular" type="xsd:int"/>
     <wsdl:part name="telefone_fixo" type="xsd:int"/>
   </wsdl:message>
  ▼<wsdl:message name="PesquisarServicoRequest">
     <wsdl:part name="atividade" type="xsd:string"/>
     <wsdl:part name="cidade" type="xsd:string"/>
   </wsdl:message>
   <wsdl:message name="CadastrarUsuarioResponse"></wsdl:message>
   <wsdl:message name="CadastrarServicoResponse"></wsdl:message>
  ▼<wsdl:message name="EnviarEmailRequest">
     <wsdl:part name="email_indicado" type="xsd:string"/>
     <wsdl:part name="texto" type="xsd:string"/>
   </wsdl:message>
```

Figura 16. Serviço criado pelo Axis.

Como mostra na Figura **16**, a classe *Servicos.java* foi renomeada para Servico.jws e colocado dentro da pasta de instalação do Axis, tornando assim um serviço web com seus métodos públicos, gerando o arquivo WSDL e com o endereço do serviço *http://localhost:8081/axis/Servicos.jws?wsdl*.

Na Figura acima observa-se que todos os serviços possuem elementos de entrada. Quando invocado o serviço de *CadastrarUsuario*, esses elementos devem conter informações vinda das aplicações clientes para serem repassadas ao método *CadastrarUsuario* que foi implementado na Figura **15** e assim, posteriormente, essas informações poderão ser inseridas no banco de dados. Dessa mesma maneira, foram desenvolvidos todos os serviços necessários para a utilização da arquitetura voltada para prestadores de serviços autônomos.

Assim, foi concluído o desenvolvimento dos serviços e do *Web Service* onde esses serviços estão armazenados. A próxima seção descreverá um sistema web de teste construído com o objetivo de consumir esses serviços criados.

5.5.3 Criação do sistema Web para consumo dos serviços

Como proposto neste trabalho, foi desenvolvido uma aplicação web de teste para o consumo dos serviços disponibilizados no Web Service, a Figura **17** ilustra como o projeto está estruturado:

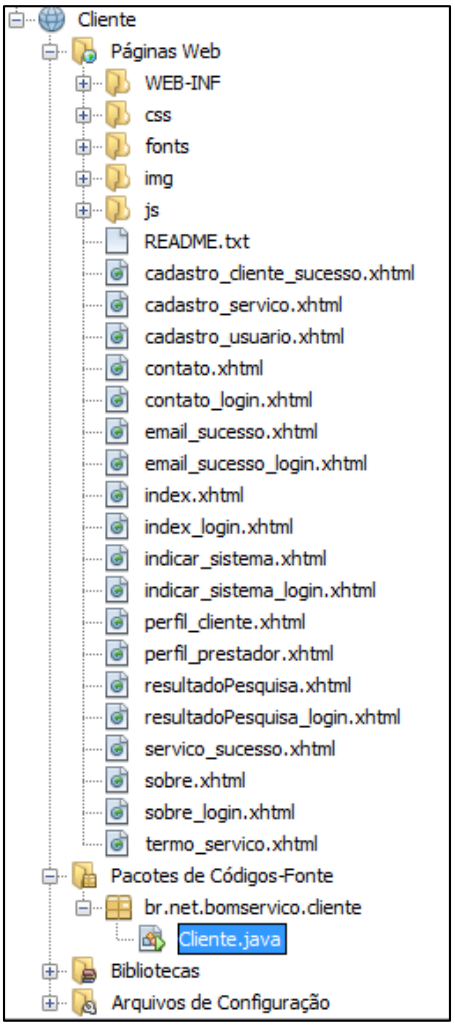


Figura 17. Estrutura do projeto Web.

Na Figura **17** é apresentado a estrutura de pastas, dependências e configurações de todo o projeto Web necessário para o consumo dos serviços desenvolvidos citados na seção 5.5.1 deste trabalho.

Na pasta *Páginas Web* contém documentos de configuração do projeto, subpastas de imagens e arquivos eXtensible Hypertext Markup Language (XHTML) que proporciona a padronização de exibição de páginas Web em diversos dispositivos.

O pacote *br.net.bomservico.cliente* encontra-se a classe responsável por fazer acesso aos serviços disponibilizados pelo *Web Service*. A Figura **18** demonstra como é realizada a chamada dinâmica dos serviços disponibilizado através do *web service*.

```
3   import org.apache.axis.client.Service;
4   import org.apache.axis.client.Call;
5
6   public class Cliente {
7
8       // Endereço, local onde encontra-se o Web Service
9       String local = "http://localhost:8081/axis/Servico.jws";
10      // Criando e configurando o serviço
11      Call call = (Call) new Service().createCall();
12
13      // Configurando o endereço.
14      call.setTargetEndpointAddress (local);
15
16      // Método a ser chamado.
17      public void cadastrarUsuario() {
18          call.setOperationName("CadastrarUsuario");
19          // Parâmetros da função CadastrarUsuario.
20          Object[] param = new Object[]{new String(), new String(), new String(), new String()};
21          // Retorno da Função
22          call.invoke(param);
23      }
24
25      // Método a ser chamado.
26      public void PesquisarServico() {
27          call.setOperationName("PesquisarServico");
28          // Parâmetros da função PesquisarServico.
29          Object[] param = new Object[]{new String(), new String()};
30          // Retorno da Função
31          call.invoke(param);
32      }
33  }
```

Figura 18. Classe responsável pelo consumo dos serviços.

Como é possível verificar, na linha 9 da Figura **18**, é criada uma variável que recebe o caminho do *web service*, na linda 11 é criado um objeto *Call* recebendo um novo serviço e por fim na linha 14 é feito a configuração do endereço, pronto a parte de configuração do serviço está concluída, na linha 18 é definido qual serviço será consumido e na linha 20 é criado um *array* para acomodar os parâmetros do serviço, a imagem exibe dois métodos disponíveis para consumo.

A Figura **19** apresenta a tela inicial do sistema, onde é possível fazer uma pesquisa de acordo com sua região, cidade e serviço que deseja encontrar.

Figura 19. Tela principal da aplicação.

Como ilustrado na Figura **19**, é possível visualizar o menu com indicações a amigos, conhecer sobre a ferramenta, enviar e-mails aos administradores do sistema, efetuar login e cadastra-se. Só será possível efetuar uma pesquisa se o cliente selecionar o estado, cidade e atividade que deseja pesquisar.

5.5.4 Recursos

Com base em alguns requisitos elicitados os recursos mais importantes serão detalhados nesta seção: cadastrar usuário, pesquisar prestador de serviço e cadastrar serviço. Nas subseções que cada um será abordado.

5.5.4.1 Login/Cadastrar Usuário

Este recurso fornece ao usuário a opção de acessar e/ou cadastrar-se para que possa desfrutar de suas funcionalidades. A Figura **20** apresenta a interface gráfica de usuário usada para este recurso:

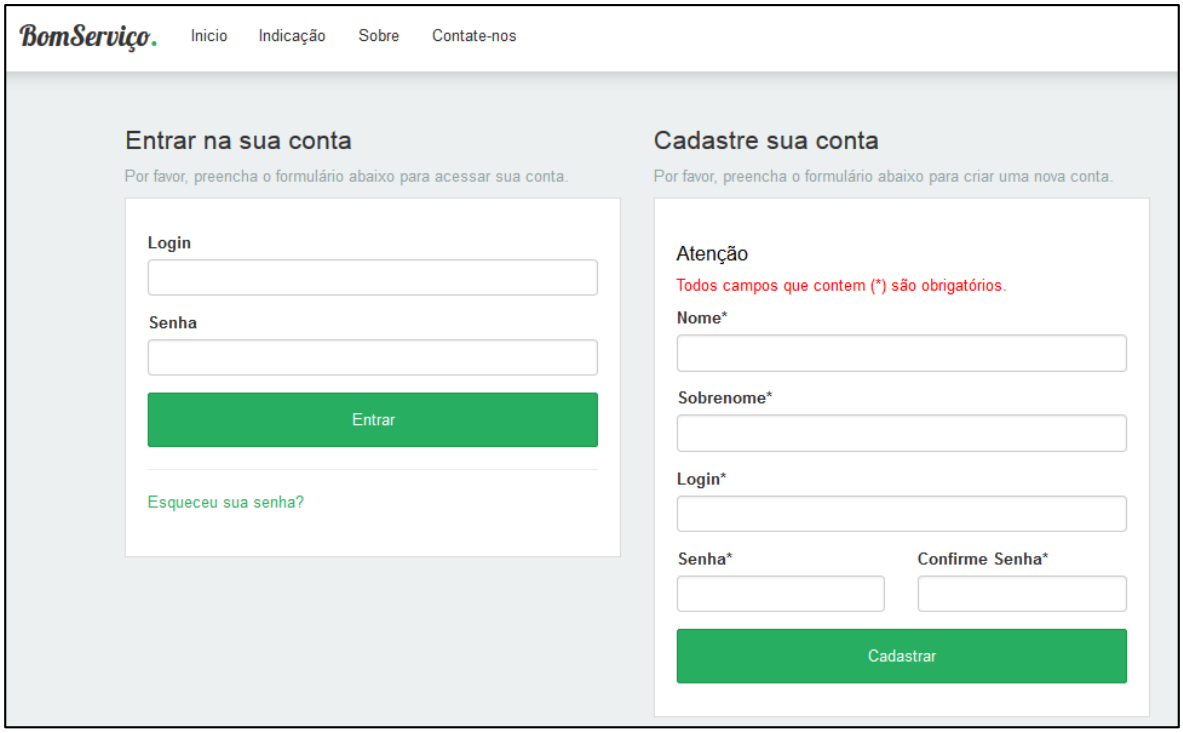


Figura 20. Tela de login/cadastro de usuários.

Quando a tela de cadastro de usuário é acionada, e exibido duas opções:

1. **Entrar na sua conta:** caso já tenha efetuado o cadastro basta preencher o formulário com seu login e senha para acessar o sistema;
2. **Cadastrar usuário:** exige que seja preenchido um formulário para criar uma conta com os seguintes dados: nome; sobrenome; login; senha e confirma senha;

Após clicado no botão de cadastrar caso não haja nenhum erro, o sistema armazenará seus dados habilitando a utilização por completo da ferramenta.

5.5.4.2 Pesquisar Prestador de Serviço

Pesquisar prestadores de serviços é um recurso que provê ao usuário sugestões de prestadores mais próximo de sua localidade. A interface responsável por apresentar o recurso pode ser visualizada na Figura **21**:

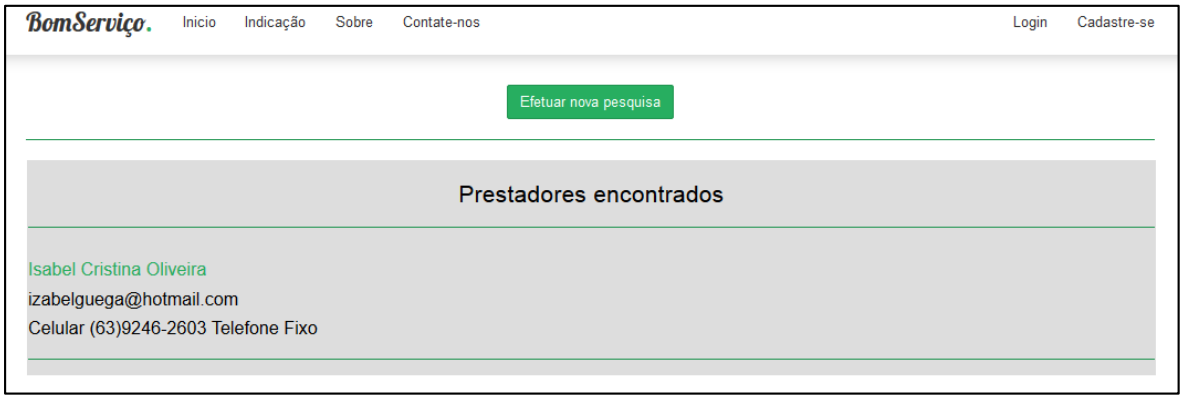


Figura 21. Tela de Prestadores encontrados na pesquisa.

Como pode ser visto, o recurso informa um prestador de serviço de acordo com a atividade pesquisada, com nome completo, e-mail para contato e telefone.

5.5.4.3 Cadastrar Serviço

Este recurso possibilita que um prestador de serviço cadastrado na base de dados possa expor seus serviços na internet de forma rápida e fácil, abrangendo uma área bem maior que outros tipos de divulgação. A interface responsável por apresentar o recurso pode ser visualizada na Figura **22**:

Figura 22. Tela de castrado de serviço.

Com base na imagem acima, para efetuar o cadastro de um serviço é necessário está devidamente cadastrado no sistema e preencher o formulário com: profissão exercida; descrição do serviço ofertado; número de telefone celular para contato; valor cobrado pelo serviço; estado em que oferta o serviço e cidade.

6 CONSIDERAÇÕES FINAIS

Descrever Considerações finais

7 REFERÊNCIAS BIBLIOGRÁFICAS

IBGE. **Instituto Brasileiro de Geografia e Estatística**. Acessado em 25 de Agosto de 2014. Disponível em: <http://www.ibge.gov.br/home>.

LUGER, GEORGE F. (2004). **Inteligência Artificial: Estruturas e Estratégias para a Solução de Problemas Complexos**. (4ª ed.), Porto Alegre: Bookman.

BOOCH, G. and RUMBAUGH, J. and JACOBSON. (2006). **UML: Guia do usuário** (2ª ed.). Campus – RJ.

MARZULLO, F. P. (2009). **SOA na Prática**. Novatec.

KUMAR, B. V. (2012). **Implementando SOA Usando JAVA™ EE**. Rio de Janeiro: ALTA BOOKS.

SHARP, J. (2011) **Microsoft Visual C# 2010: passo a passo**. Tradução de Tereza Cristina Félix de Sousa e Edson Furmankiewicz. Porto Alegre: Bookman.

GETNINJAS – (2011) – **Aplicação web de mesmo contexto.** Disponível em: <http://www.getninjas.com.br/>

RECOMIND.NET – (2011) – **Aplicação web de mesmo contexto.** Disponível em: <http:// http://www.recomind.net/>

SAUDATE, A. (2013). **SOA aplicado: Integrando com web services e além**. São Paulo: Casa do Código.

ERL, T. (2011). **Introdução às tecnologias Web Services: SOA, SOAP, WSDL e UDDI**. WebMobile.

CALIENDO, R. F. (2010). **Desenvolvimento de uma Aplicação utilizando SOA: um Estudo de Caso**. pp. 2-9.

GUEDES, G.T.A. (2011). **UML 2 - UMA ABORDAGEM PRÁTICA** (2 ed.). São Paulo: NOVATEC.

RAMEZ, E.; NAVATHE, S. B. **Sistemas de banco de dados**. 4ª. ed. São Paulo: Pearson Addison Wesley, 2005.

FIETO. **Federação das Indústrias do Estado do Tocantins**. Acessado em 28 de Outubro de 2014. Disponível em: < http://www.fieto.com.br/DownloadArquivo.aspx?c=0386b5a3-7164-4170-b69c-1000b1b6d30b >

SOMMERVILLE, I. (2011). Engenharia de Software (9 ed.). (I. Bosnic, & K. G. de O. Gonçalves, Trads.) São Paulo: Pearson Prentice Hall.

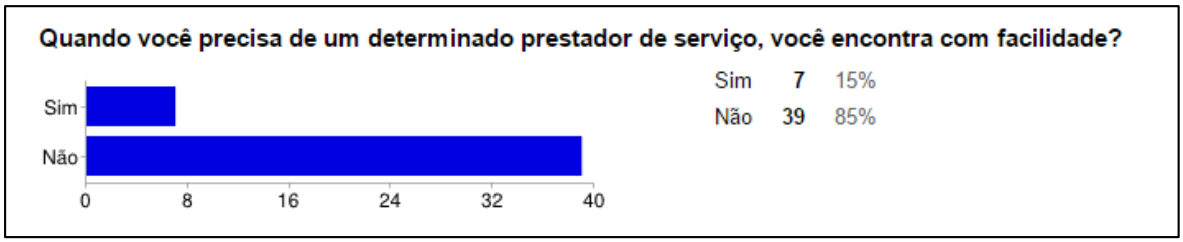
DORNELAS, J. C. A. **Empreendedorismo**: transformando ideias em negócios. 2 ed., Rio de Janeiro: Elsevier, 2005.

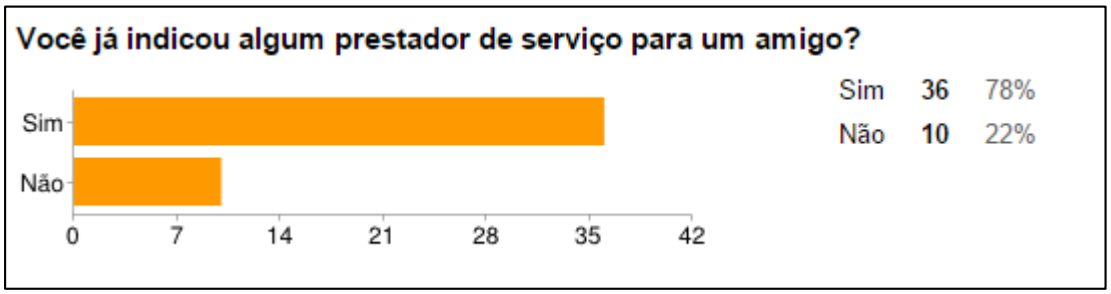
SEBRAE. **Anuário do trabalho na micro e pequena empresa**. 6. ed. Distrito Federal: Dieese, 2013.

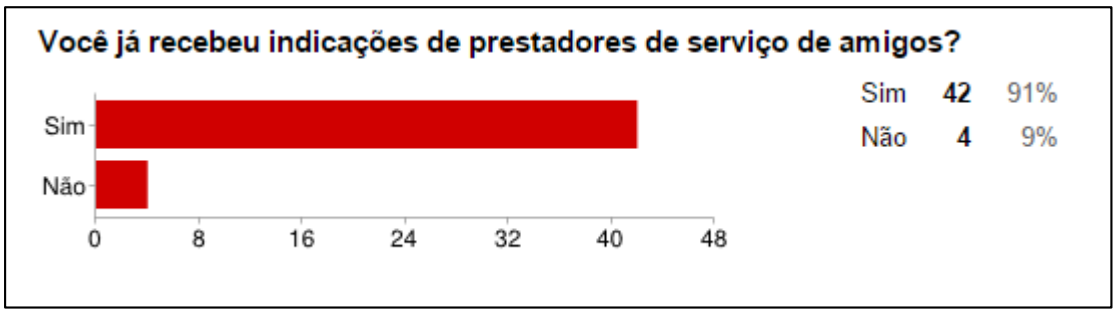
OSTERWALDER, A.; PGNEUR, Y. **Business Model Generation**: Inovação em Modelos de Negócios. Rio de Janeiro: Alta Book, 2011.

APÊNDICE A

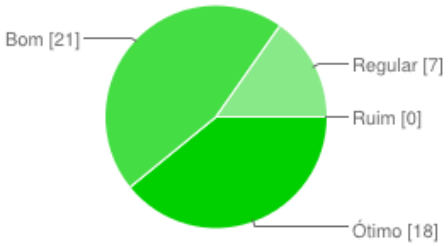
Resultados obtidos na pesquisa de coleta de dados.







O que você acha desse método de contratação de prestadores de serviços através da internet?



Ótimo	18	39%
Bom	21	46%
Regular	7	15%
Ruim	0	0%

Se houvesse um aplicativo oferecendo prestadores de serviços de acordo com sua região, você usaria?

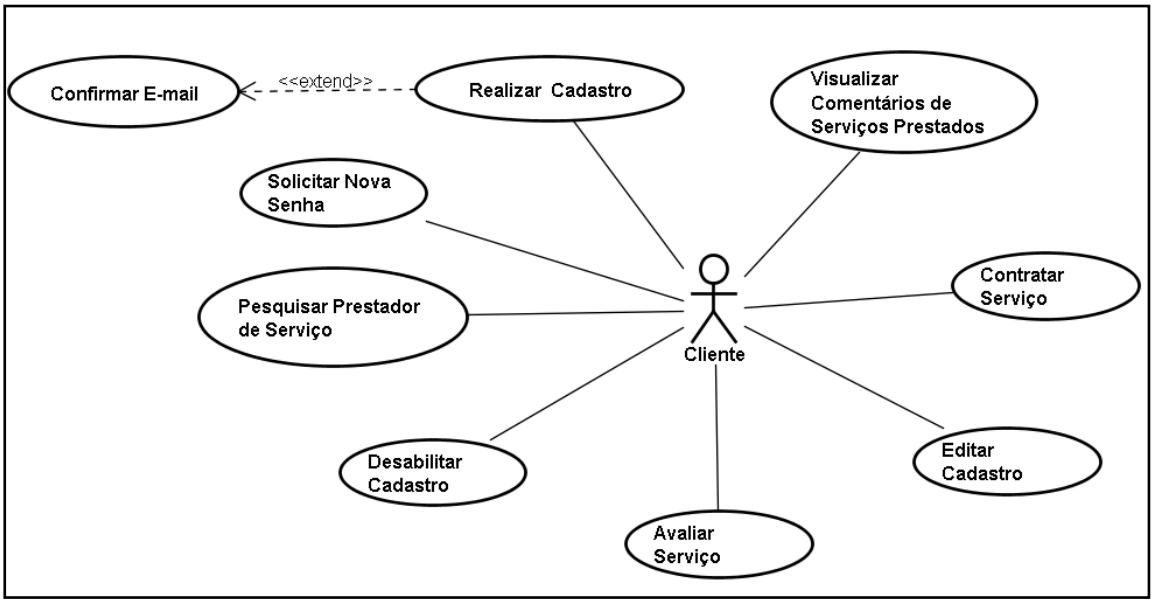
Sim	44	96%
Não	2	4%

Se houvesse um site em que você pudesse expor seu serviço você usaria?

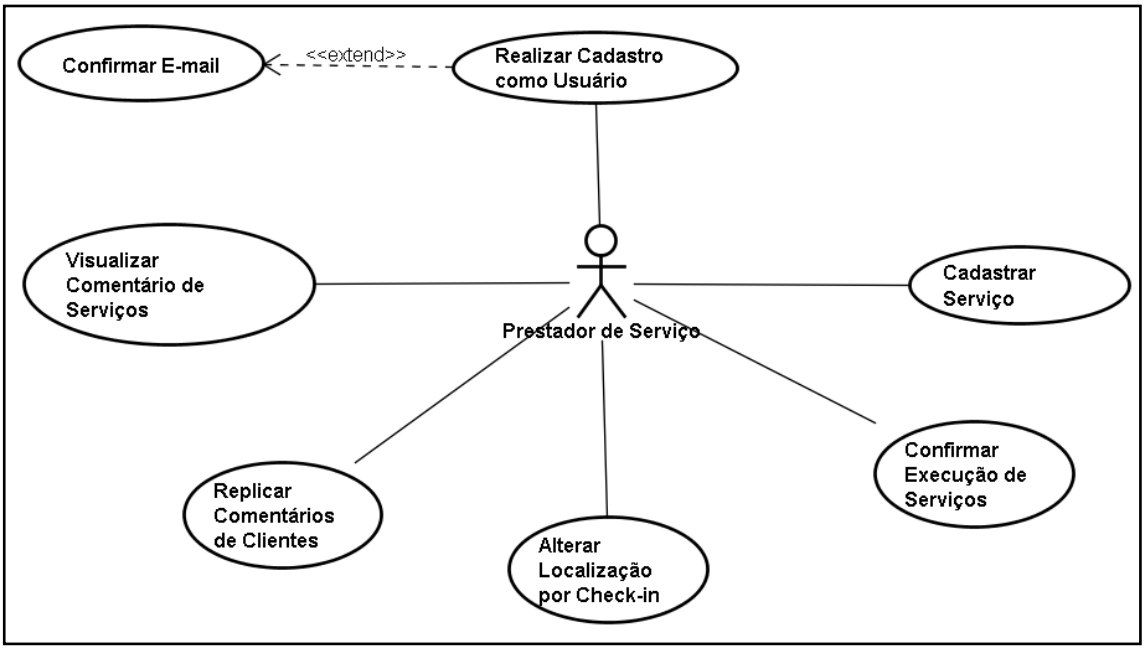
Sim	37	80%
Não	2	4%

APÊNDICE B

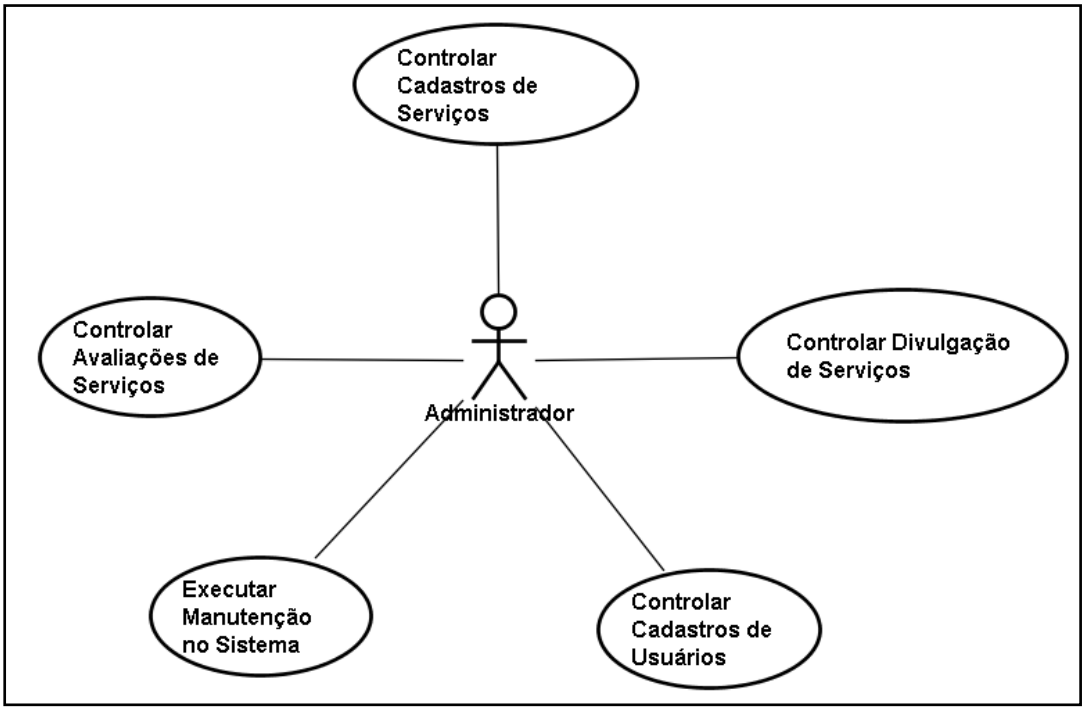
Caso de Uso do Cliente



Caso de Uso do Prestador de Serviço



Caso de Uso do Administrador



Principais Casos de Uso Expandidos

Cadastrar Usuário

Caso de Uso:	Cadastrar Usuário
Ator(es):	Cliente
Finalidade:	Cadastrar o usuário no sistema
Visão Geral:	Permitir que o usuário se autentique no sistema para usufruir de suas funcionalidades.
Pré-Condições:	-
Pós-Condições:	Redirecionar o usuário para a tela principal da aplicação

Sequência Típica de Eventos	
Ação do Ator	**Resposta do Sistema**
1. Insere dados.	
	2. Validação de dados. Ser for validado com sucesso é redirecionado a página principal.

Sequência Alternativa	
Linha 2	Se não ocorrer sucesso na validação, o sistema exibe uma mensagem para o usuário.

Cadastrar Serviço

Caso de Uso:	Cadastrar Serviço
Ator(es):	Prestador de Serviço
Finalidade:	Cadastrar um serviço no sistema
Visão Geral:	Permitir que o usuário prestador de serviço possa cadastrar um serviço no sistema.
Pré-Condições:	Ter o usuário cadastrado no sistema.
Pós-Condições:	Redirecionar o usuário para a tela principal da aplicação, e disponibilizar seu serviço para os demais clientes da aplicação.

Sequência Típica de Eventos	
Ação do Ator	**Resposta do Sistema**
1. Insere dados.	
	2. Validação de dados. Ser for validado com sucesso é redirecionado a página principal.

Sequência Alternativa	
Linha 2	Se não ocorrer sucesso na validação, o sistema exibe uma mensagem para o prestador de serviço.

Pesquisar Serviço

Caso de Uso:	Pesquisar Serviço
Ator(es):	Cliente
Finalidade:	Pesquisar um serviço no sistema.
Visão Geral:	Permitir que o cliente possa pesquisar um prestador de serviço cadastrado no sistema.
Pré-Condições:	Ter o serviço cadastrado no sistema.
Pós-Condições:	Exibir resultado da pesquisa para o cliente.

Sequência Típica de Eventos	
Ação do Ator	**Resposta do Sistema**
1. Informa o estado, cidade e atividade.	
	2. Exibir resultado da pesquisa.

Sequência Alternativa

Linha 2	Nenhum serviço encontrado.

Controlar Cadastros de Usuários

Caso de Uso:	Controlar Cadastro de Usuários
Ator(es):	Administrador do Sistema.
Finalidade:	Controlar todos os usuários cadastrados no sistema.
Visão Geral:	Garantir autenticidade dos dados dos usuários cadastrados no sistema.
Pré-Condições:	Usuários cadastrados.
Pós-Condições:	-

Sequência Típica de Eventos	
Ação do Ator	**Resposta do Sistema**
1. Pesquisar todos os usuários cadastrados no sistema.	
	2. Exibir resultado da busca do administrador.
3. Efetuar um contato com o prestador para verificação dos dados cadastrados.	
	4. Contato efetuado com sucesso.

Sequência Alternativa	
Linha 2	Nenhum serviço encontrado.
Linha 3	Falha ao efetuar contato com o prestador de serviço.

APÊNDICE C

Dicionário de Dados

Abaixo a descrição de cada tabela contendo o dicionário de dados.

Tabela 1: endereco				
Chave Primaria	**Nome**	**Tipo**	**Tamanho**	**Descrição**
PK	idendereco	INT	-	Chave primaria da tabela endereco, autoincremento tem por finalidade identificar a posição dos dados cadastrados.
	cep	INT	-	Campo responsável por armazenar o cep do endereço.
	quadra	VARCHAR	45	Campo responsável por armazenar o nome da quadra do endereço.
	rua	VARCHAR	45	Campo responsável por armazenar o nome da rua do endereço.
	numero	VARCHAR	45	Campo responsável por armazenar o número do endereço.
	complemento	VARCHAR	45	Campo responsável por armazenar algum ponto de referência para fácil identificação do seu endereço.
	setor	VARCHAR	45	Campo responsável por armazenar o nome do setor do endereço.
FK	fk_idcidade	INT	-	Campo chave estrangeira contem a referência do id da tabela cidade.
Obs.: essa tabela surgiu através do requisito RFC0002 descrito na seção 5.2				

Tabela 2: cidade				
Chave Primaria	**Nome**	**Tipo**	**Tamanho**	**Descrição**
PK	idcidade	INT	-	Chave primaria da tabela cidade, autoincremento tem por finalidade identificar a posição dos dados cadastrados.
	cidade	VARCHAR	60	Campo responsável por armazenar o nome da cidade.
FK	fk_uf	INT	2	Campo chave estrangeira tem finalidade de pegar o id do estado que está na tabela de uf.
Obs.: essa tabela surgiu através do requisito RFC0002 descrito na seção 5.2				

Tabela 3: uf				
Chave Primaria	**Nome**	**Tipo**	**Tamanho**	**Descrição**
PK	iduf	VARCHAR	2	Chave primaria da tabela uf, tem por finalidade identificar a posição dos dados cadastrados.
	estado	VARCHAR	45	Campo responsável por armazenar o nome do

				estado.
Obs.: essa tabela surgiu através do requisito RFC0002 descrito na seção 5.2				

Tabela 4: usuario				
Chave Primaria	**Nome**	**Tipo**	**Tamanho**	**Descrição**
PK	idusuario	INT	-	Chave primaria da tabela usuario, autoincremento tem por finalidade identificar a posição dos dados cadastrados.
	login	VARCHAR	45	Campo responsável por armazenar o login de um usuário, necessário para ser identificado no sistema e ter acesso as funcionalidades do mesmo.
	nome	VARCHAR	45	Campo responsável por armazenar o nome do cliente.
	sexo	VARCHAR	1	Campo responsável por armazenar o sexo da pessoa apenas com uma letra M para masculino ou F para feminino.
	email	VARCHAR	30	Campo responsável por armazenar o e-mail do cliente.
	senha	VARCHAR	20	Campo responsável por armazenar a senha do usuário, para ter acesso ao sistema.
	cpf	INT	-	Campo responsável por armazenar o cpf do prestador de serviço.
	foto	VARCHAR	255	Campo responsável por armazenar o caminho da foto do perfil do prestador de serviço no banco de dados.
	celular	INT	-	Campo responsável por armazenar o número do celular do prestador de serviço.
	telefone_fixo	INT	-	Campo responsável por armazenar o número de telefone fixo do prestador de serviço.
	data_cadastro	TIMESTAMP	-	Campo responsável por armazenar a data do cadastro do usuário. O campo Default está definido como now(), método responsável por pegar a data e hora exata do cadastro.
FK	fk_idendereco	INT	-	Campo que contem a referência do id da tabela endereco.
Obs.: essa tabela surgiu através dos requisitos RFC0001, RFC0002 descrito na seção 5.2				

Tabela 5: servico				
Chave Primaria	**Nome**	**Tipo**	**Tamanho**	**Descrição**
PK	idservico	INT	-	Chave primaria da tabela servico, autoincremento tem por finalidade identificar a posição dos dados cadastrados.
	atividade	VARCHAR	45	Campo responsável por armazenar o nome do serviço ofertado.
	descricao	VARCHAR	255	Campo responsável por armazenar uma descrição do serviço informado.
	site	VARCHAR	45	Campo responsável por armazenar o endereço virtual do prestador de serviço como: site, facebook, twitter etc.
	data_cadastro	TIMESTAMP	-	Campo responsável por armazenar a data do cadastro do serviço. Default está definido como now(), método responsável por pegar a data e hora exata do cadastro.
	valor_servico	FLOAT	-	Campo responsável por armazenar o valor do serviço ofertado.
	situação_servico	INT	1	Campo responsável por armazenar qual a situação que o serviço se encontra exemplo: em aberto, finalizado, em andamento.
FK	fk_idusuario	INT	-	Campo que contem a referência do id da tabela usuario.
Obs.: essa tabela surgiu através do requisito RFC0002 descrito na seção 5.2				

Tabela 6: avaliacao_servico				
Chave Primaria	**Nome**	**Tipo**	**Tamanho**	**Descrição**
PK	idavaliacao_servico	INT	-	Chave primaria da tabela avaliacao_servico, autoincremento tem por finalidade identificar a posição dos dados cadastrados.
	comentario	VARCHAR	255	Campo responsável por armazenar comentários sobre um determinado serviço.
	estrela	INT	-	Campo responsável por armazenar estrelas como outra forma de avaliação.
	data_avaliacao	TIMESTAMP	-	Campo responsável por armazenar a data do cadastro da avaliação do serviço. Default está definido como now(), método responsável por pegar a data e

				hora exata do cadastro.
FK	fk_idservico	INT	-	Campo chave estrangeira contem a referência do id da tabela serviço.
Obs.: essa tabela surgiu através dos requisitos RFC0010 e RFC0013 descrito na seção 5.2				

Tabela 7: servico_tem_categoria_servico				
Chave Primaria	**Nome**	**Tipo**	**Tamanho**	**Descrição**
PK	idservico	INT	-	Chave primaria da tabela servico, autoincremento tem por finalidade identificar a posição dos dados cadastrados.
PK	idcategoria_servico	INT	-	Chave primaria da tabela categoria_servico, autoincremento tem por finalidade identificar a posição dos dados cadastrados.
Obs.: essa tabela surgiu através da ligação de muitos para muitos entra a tabela servico e categoria_serviço descrito na seção 5.4.4.				

Tabela 8: categoria_servico				
Chave Primaria	**Nome**	**Tipo**	**Tamanho**	**Descrição**
PK	idcategoria_servico	INT	-	Chave primaria da tabela categoria_servico, autoincremento tem por finalidade identificar a posição dos dados cadastrados.
	descicao	VARCHAR	45	Campo responsável por armazenar uma descrição da categoria do serviço.
	status	INT	1	Campo responsável por armazenar um status da categoria do serviço, informação necessária para administradores do sistema, para que possa atribuir categorias aos serviços cadastrados.
Obs.: essa tabela surgiu através do requisito RFC002 descrito na seção 5.2				