Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

VINÍCIUS JOSÉ SILVA DOS SANTOS

DESENVOLVIMENTO DE APLICAÇÕES GOOGLE ANDROID E INTEGRAÇÃO COM WEB SERVICE

ASSIS
2012

Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

VINÍCIUS JOSÉ SILVA DOS SANTOS

DESENVOLVIMENTO DE APLICAÇÕES GOOGLE ANDROID E INTEGRAÇÃO COM WEB SERVICE

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito de Curso de Graduação.

Orientador: Dr. Almir Rogério Camolesi
Co-Orientador: Esp. Guilherme de Cleva Farto
Área de Concentração: Informática

ASSIS
2012

FICHA CATALOGRÁFICA

SANTOS, Vinícius J.S.
Desenvolvimento de aplicações Google Android e integração com web service/ Vinícius José Silva dos Santos. Fundação Educacional do Município de Assis - FEMA – Assis, 2012.
48p.

Orientador: Almir Rogério Camolesi
Trabalho de Conclusão de Curso – Instituto Municipal de Ensino Superior de Assis – IMESA.

1.Google Android 2.SOA 3.Web Service

CDD: 001.6
Biblioteca da FEMA

Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

VINÍCIUS JOSÉ SILVA DOS SANTOS

DESENVOLVIMENTO DE APLICAÇÕES GOOGLE ANDROID E INTEGRAÇÃO COM WEB SERVICE

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação, analisado pela seguinte comissão examinadora:

Orientador: Dr. Almir Rogério Camolesi
Co-Orientador: Esp. Guilherme de Cleva Farto
Analisador: Esp. Diomara Martins Reigato Barros

Assis
2012

DEDICATÓRIA

Dedico este trabalho a minha família no qual me proporcionou uma ótima base para que pudesse realizar meus objetivos, e aos meus amigos que durante todo esse período sempre estiveram comigo e me apoiaram, compartilhando momentos de alegria e tristeza.

AGRADECIMENTO

Agradeço aos meus pais Laura e José Ap. dos Santos, por proporcionar uma ótima estrutura, sacrificando seus objetivos para que pudesse realizar os meus.

Ao orientador Almir Rogério Camolesi e ao Co-Orientador Guilherme de Cleva Farto por todo apoio e conselho oferecido para a elaboração, quanto na minha vida acadêmica.

Aos meus grandes amigos, que sempre estiveram comigo em momentos importantes e nas horas mais difíceis, dando apoio e confiança para realizar meus objetivos.

E a todos que de alguma forma contribuíram diretamente ou indiretamente para a realização do trabalho.

“O sucesso nasce do querer, da determinação e persistência em se chegar a um objetivo. Mesmo não atingindo o alvo, quem busca e vence obstáculos, no mínimo fará coisas admiráveis.”

José de Sousa Saramago

(1922 – 2010).

RESUMO

O desenvolvimento deste projeto foi de pesquisar e abranger os conhecimentos sobre arquitetura orientada a serviços, demonstrando na teoria e na pratica como ela pode ser empregada na forma de um *Web Service*, como também no desenvolvimento do aplicativo na plataforma *Google Android* para consumir esses serviços disponibilizados na web.

Designado especificamente para o uso das Assistentes Sociais, este sistema permite a consulta via web toda vez que necessário, a fim de proporcionar rapidez e comodidade no seu dia-a-dia.

Palavras Chave: Google Android, SOA, Web Services.

ABSTRACT

The development of this project was to research and cover the knowledge about service-oriented architecture, demonstrating in theory and practice how it can be used as a Web Service, as well as the application development platform for Google Apps consume these services available web. Specifically designated for the use of Social Workers, this system allows the query via web whenever necessary in order to provide speed and convenience in their day-to-day.

Keywords: Google Android, SOA, Web Services.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

XML	Extensible Markup Language
GPS	Global Positioning System
ERP	Enterprise Resourse Planning
ESB	Enterprise Service Bus
HTTP	Hypertext Transfer Protocol
W3C	World Wide Web Consortium
WSDL	Web Services Description Language
SOAP	Simple Object Access Protocol
UDDI	Universe Description Discovery and Integration
JDOM	Java Document Object Model
DOM	Document Object Model
SAX	Simple API for XML
HQL	Hibernate Query Language
SQLs	Structured Query Language
OHA	Open Handset Alliance

SUMÁRIO

1. INTRODUÇÃO

Há vinte anos, o telefone móvel era um aparelho arcaico, apenas servia para fazer ligações, sobretudo era visto na sociedade como um ícone de status social, pois uma habilitação da linha naquela época custava em média US$ 20 mil dólares e o aparelho US$ 3,5 mil dólares, ao passar dos anos ela se popularizou no País, com a chegada das famosas linhas pré-pagas, no qual até hoje é a preferência dos brasileiros com 82,23% do mercado brasileiro (BARBOSA, 2011).

Segundo os dados da Agência Nacional de Telecomunicações (ANATEL), existem mais linhas habilitadas de celulares do que habitantes no País. O Brasil fechou o mês de fevereiro de 2011, com 207,5 milhões de acessos móveis de telefonia celular (ESTADÃO, 2011).

De olho no mercado superaquecido, muitas organizações corporativas estão buscando integrar suas aplicações com sistemas móveis, otimizando o processo de negócio, a fim de tornar a mobilidade uma fonte de lucros, para realizar essa sincronização de informações de um servidor web a um aplicativo é necessário uma tecnologia que faça essa comunicação, existem alguns recursos dentre elas, a mais utilizada são Web Services, que por sua vez, permite a vários sistemas comunicarem com seu servidor, assim criando um serviço na internet de forma interativa.

1.1 OBJETIVOS

Objetivo principal desse trabalho é pesquisar e abranger o conhecimento sobre a arquitetura orientada a serviços, como a utilização de Web Services, a fim de desenvolver uma solução para dispositivos móveis, usando a plataforma *Android*.
No desenvolvimento do aplicativo móvel, será realizada a integração de um módulo do Sistema REDECA, desenvolvida pela Fundação Telefônica em parceria com oito municípios paulistas, tendo a finalidade de organizar, otimizar e aprimorar a qualidade do sistema de atendimento infanto-juvenil, interagindo com a API *Google Maps*, no qual contem funcionalidade de pesquisa e visualização de mapas e

imagens de satélite, com o *Global Positioning System* (GPS), que fornece a localização exata de um ponto na superfície terrestre.

1.2 JUSTIFICATIVAS E MOTIVAÇÕES

Com o crescimento do mercado móvel, a mobilidade passou a ser uma alternativa rentável para as grandes empresas corporativas, a fim de aperfeiçoar os processos de negócios e agilizar as tomadas de decisões, buscando integrar suas aplicações móveis com sistemas distribuídos existentes, entretanto, a uma necessidade de um estudo acadêmico sobre o tema, aonde os estudantes possam entender e utilizar como base em seus trabalhos futuros.

Com o amplo avanço na procura por softwares de localização via satélite, o usuário cada vez mais busca comodidade e nada melhor do que atrelar tecnologia com praticidade, ao pesquisar vários aplicativos que utilizam a tecnologia GPS, é notória a falta de um aperfeiçoamento do mesmo, com mais recursos, como encontrar pontos de interesses do usuário, extrair informações de velocidade do trafego, a fim de orientar ao usuário uma rota alternativa.

Outra motivação é o mercado emergente para desenvolvedores de aplicativos móveis, pela grande necessidade de mão-de-obra qualificada.

1.3. ESTRUTURA DO TRABALHO

Este trabalho foi organizado em seis capítulos, sendo o primeiro esta introdução.

No segundo capítulo, será apresentado o conceito de SOA – Arquitetura orientada a serviços.

No terceiro capítulo, serão apresentadas as fundamentações teóricas sobre Web Services.

No quarto capítulo, serão apresentados o conceito e a arquitetura da plataforma Google Android.

No quinto capítulo apresenta o desenvolvimento do trabalho, mostrando a modelagem do problema, casos de uso e os resultados obtidos.

No sexto capítulo apresenta a conclusão do trabalho.

2. SOA –ARQUITETURA ORIENTADA A SERVIÇOS

Neste capitulo iremos abordar o conceito de SOA (*Service-Oriented Architecture*), seus objetivos e vantagens na implementação de uma arquitetura de um sistema e sua integração com os demais sistemas, propiciando informações essenciais para a tomada de decisão.

2.1. INTRODUÇÃO

Com as constantes transformações ocorridas nas ultimas décadas, as grandes organizações precisaram se manter focadas em continuar realizando seus trabalhos, visando obter vantagem competitiva e a captação de novos clientes, as empresas careciam de inovação, criatividade, lançar novos produtos e serviços com qualidade, objetivando sempre o primeiro lugar.

A partir desse momento as grandes empresas, começaram a levar em consideração o uso da informação, com a criação de sistemas integrados de gestão, por exemplo, os ERP (*Enterprise Resourse Planning*), poderosíssima ferramenta no qual tem a finalidade de coletar e armazenar dados, a fim de gerar informações substanciais no auxilio em suas estratégias e na tomada de decisões.

O número de sistemas heterogêneos que existem dentro de uma empresa, tem crescido sem precedentes, aonde muitos sistemas são desenvolvidos com o propósito de atender as especificações distintas de cada um dos setores, isto acaba resultando em aplicações que não estão aptas a trocarem informações (RABELO, 2006).

Com o passar dos tempos, os processos e os sistemas se tornaram cada vez mais complexos e inflexíveis, antigamente era possível tratar alguns problemas de escalabilidade e distribuição com a harmonização, que seria intervenções manuais ou de soluções de código, só que essa manutenibilidade ao decorrer dos anos acaba se tornando custoso e demorado demais, observando essa necessidade de se manterem no mercado bastante concorrido, as organizações notaram-se obrigados a se tornarem ágeis e flexíveis. Estas aplicações carecem em se comunicar de forma conjunta com a finalidade de atingir agilidade e simplificar processos de negócio, tornando-os mais produtivos. A solução, entretanto, não é

desmontar e substituir sistemas ou aplicativos, nem renová-los completamente, mas descobrir uma forma de potencializar investimentos em TI (Tecnologia de Informação) existentes para que as metas organizacionais gerais sejam lidadas de maneira hábil (MICROSOFT, 2011).

Com a necessidade de obter integração fez surgir novas formas de abordagem da integração de sistema, como SOA, que vem apontando como o mais novo paradigma de desenvolvimento de sistemas, representando uma nova metodologia de pensar quanto ao projeto da arquitetura de um sistema e sua posterior integração a outros sistemas.

A orientação a serviços auxilia a alcançar essas metas tornando os sistemas mais responsivos às necessidades de negócios, mais simples de desenvolver, e mais fáceis de manter e gerenciar. A orientação a serviços separa os recursos de TI em módulos, criando processos de negócios interligados e que integram informações entre sistemas de negócios, assim, os serviços seriam pequenos fragmentos de software, desenvolvidos de tal modo, que possam ser acoplados a outros componentes de software. Desenvolver uma arquitetura de solução baseada em orientação a serviços ajuda a organização a planejar mudanças previamente, em vez de responder de maneira reativa (MICROSOFT, 2011).

2.2. CONCEITOS E PRINCIPIOS

Embora encontrar um termo concreto para o termo SOA seja difícil, não pela quantidade de definições, mas sim pelo o contexto, níveis de abstração e vocabulário entre eles, no entanto, todas as definições concordam que SOA é um paradigma para melhorar a flexibilidade (JOSSUTIS, 2008).

SOA seria um paradigma arquitetural baseada em padrões para a concepção de uma infraestrutura de TI integrada, capaz de fornecer com rapidez as mudanças nas necessidades de negócios, no qual lidaria com processos de negócios distribuídos através de um complexo e heterogêneo cenário de sistemas novos e existentes de uma empresa em módulos integrados, simplificados, interoperáveis, e altamente flexíveis que podem ser modificados, aperfeiçoados e reutilizados para apoiar diretamente as metas comerciais.

Os principais conceitos técnicos de SOA são serviços, interoperabilidade através de um barramento corporativo de serviços e acoplamento fraco.

2.2.1 **Serviços**

Serviço é um recurso de uma funcionalidade corporativa independente. A funcionalidade pode ser simples (guardar e armazenar dados de clientes), ou complexa (um processo corporativo de um pedido do cliente). Por serviços se concentrarem no valor dos negócios de uma interface, eles ligam a lacuna entre o negocio e a TI (JOSSUTIS, 2008).

2.2.2 **Interoperabilidade**

Interoperabilidade é a capacidade de fazer diferentes sistemas se comunicarem uns com os outros, com essa finalidade utilizamos barramento corporativo de serviço (ESB), que consisti em um padrão de software usado para transferência de dados entre sistemas heterogêneos, ele utiliza uma associação de componentes onde todos seguem uma politica ou protocolo comum, como transformações de dados, roteamento (inteligente), lidar com segurança e confiabilidade, gerenciamento de serviços, monitoramento e *logging* (JOSSUTIS, 2008).

2.2.3 **Acoplamento fraco**

Acoplamento fraco é o conceito de redução de dependência do sistema, em razão dos processos corporativos serem distribuídos através de múltiplos “*backends*” (sistema que mantem os dados ou regras de negocio de um domínio específico), é de extrema importância minimizar as dependências, para que os efeitos das modificações não tenham um grande impacto, assim, permitindo que os sistemas ainda executam quando partes deles estão indisponíveis. Caso contrário, as modificações se tornam arriscadas demais, e as falhas de sistema podem quebrar o cenário de um sistema todo. Observe, no entanto, que há um preço pelo acoplamento fraco: a complexidade. Os sistemas distribuídos fracamente acoplados são mais difíceis de desenvolver e depurar, por esta razão, em uma SOA especifica o desenvolvedor tem que encontrar a quantidade certa de acoplamento fraco (JOSSUTIS, 2008).

2.3. OBJETIVOS

O objetivo de SOA é auxiliar às organizações na tomada de decisão, utilizando de vantagens tecnológicas ao realizar seus processos de negócios por meio da combinação de inovação de processos, governança eficaz que permita as áreas extrair melhores proveitos deste novo tipo de abordagem, e assim diminuindo os riscos inerentes a um projeto deste porte e estratégia de tecnologia, as quais giram em torno da definição, reutilização de serviços.

Um dos benefícios mais importantes que SOA fornece é a melhora da produtividade e agilidade tanto para o negócio quanto para TI, portanto, a redução de custos no desenvolvimento e manutenção dos sistemas abrangidos. Possui também como objetivo maximizar o reuso dos seus componentes (serviços), para que em médio prazo, a tarefa do desenvolvimento de uma aplicação seja primordialmente a tarefa da composição e coordenação dos serviços já implementados, aumentando o reuso e diminuindo o dispêndio de recursos.

3. *WEB SERVICES*

Neste capitulo iremos abordar o conceito de *Web Services*, fornecendo um melhor entendimento sobre essa tecnologia e esclarecendo algumas duvidas quanto ao seu uso.

3.1. INTRODUÇÃO

O cenário computacional expõe uma necessidade de cada vez mais obter formas de atender a demanda da distribuição da informação e do processamento das aplicações, em busca de soluções para essa realidade computacional, surgiram novas tecnologias e frameworks de desenvolvimento, permitindo uma maior integração entre os diversos aplicativos e serviços disponíveis na internet, que via troca de mensagens entre objetos computacionais em uma infra-estrutura de rede, proveem o compartilhamento dos recursos dos mesmos. Dentre as novas tecnologias que aplicam esse paradigma podemos destacar o *Web Services*.

O termo *Web Services* surgiu em 2.000, através da necessidade em padronizar a comunicação entre diferentes plataformas e linguagens de programação, baseado em uma arquitetura orientada a serviço possui como característica a ideia de um provedor e um consumidor do serviço.

Um dos motivos que tornam Web Services atrativos é o fato de este modelo ser baseado em tecnologias padrões, em particular XML (*Extensible Markup Language*) que é amplamente usado para descrição e troca de dados, *e* HTTP (*Hypertext Transfer Protocol*) que atua como transporte na comunicação entre o cliente e *Web Service*, ambos são definidos pelo W3C (*World Wide Web Consortium*).

Para mostrar a utilização de Web Services em uma situação cotidiana, será ilustrado um portal de vendas de ingressos para um show pela Internet, que necessita autenticar o crédito do comprador antes de concretizar com a venda. O sistema então acessa um serviço (*Web Service*) que atenta a todos os passos necessários à comprovação de crédito: Confere o histórico das compras efetuadas pelo cliente na empresa, checa a condição de crédito do consumidor no sistema público, etc. O *Web Service* obtém estes dados e retorna a situação de crédito deste consumidor

para o site. Porém, para tal prática, seriam necessários que as diversas empresas da área criassem seus *Web Services*, que ficariam acessível ao portal, como mostra a Figura 01 abaixo.

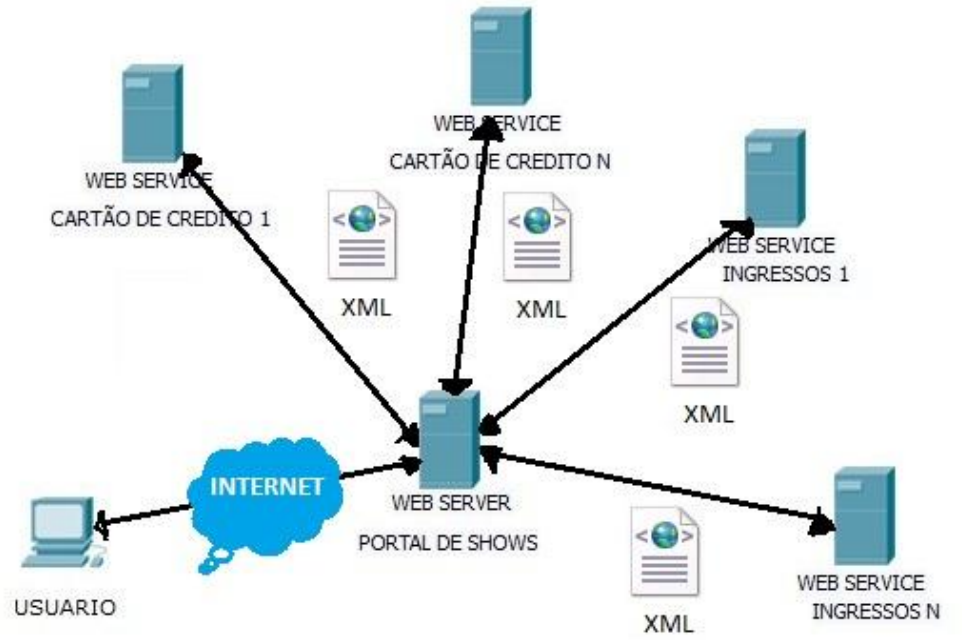

Figura 01 - *Web Services*.

3.2. OBJETIVOS

Objetivo principal do *Web Services* é estabelecer a interoperabilidade entre sistemas distribuídos heterogêneos utilizada por eles, facilitando uma melhor interligação destas aplicações. Esta interligação tem como finalidade ajudar os processos de negócios, adequando aos softwares isolados a trabalharem de forma conjunta com os demais. Assim, proporcionando um aumento na produtividade, uma maior rentabilidade e uma diminuição de custos.

Essa interoperabilidade é obtida pelos padrões abertos fundamentados em XML, o WSDL (*Web Services Description Language*) que é a linguagem de descrição dos Web Services, o SOAP (*Simple Object Access Protocol*) o protocolo usado para a

troca das informações e o UDDI (*Universe Description Discovery and Integration*) este responsável por manter uma estrutura de busca e publicação de um serviço. Estes padrões proveem uma aproximação comum definindo, publicando e consumindo os Web Services.

Além da interoperabilidade outro ponto a ser destacado é o reuso dos componentes pertencentes aos sistemas integrados, onde cada componente pode conceber um serviço distinto, podendo participar de múltiplos sistemas fornecendo maiores benefícios imediatos e aumento da agilidade do negócio (RECKZIEGEL, 2009).

3.3. ARQUITETURA

A arquitetura do *Web Service* introduz três papéis principais como base de interação: Fornecedor de Serviços, Consumidor de Serviços e Broker dos Serviços. A interação destes personagens envolve as operações de publicação, busca e ligação.

Podemos ver na Figura 02.

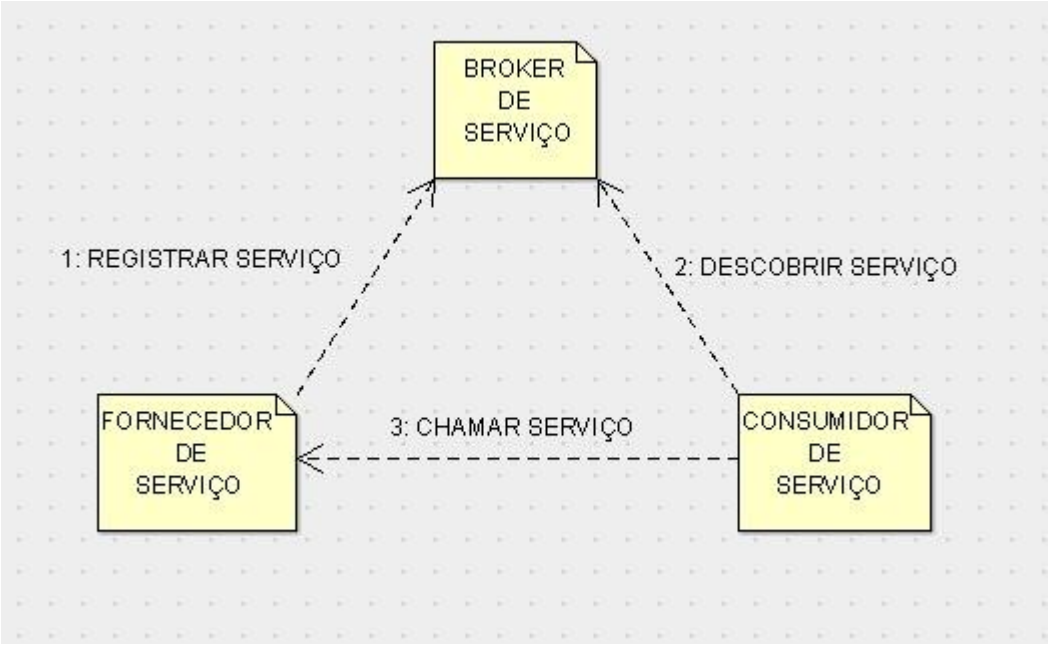

Figura 02 - Fornecedor, consumidor e *broker* (RECKZIEGEL, 2009).

3.3.1 **Fornecedor de serviços**

Constitui em uma entidade que cria e torna acessível o serviço para que alguém possa aproveitá-lo. Descrevendo o Web Service em uma forma padrão, que seja compreensível para qualquer um que necessite usar esse serviço. Publica os detalhes sobre seu Web Service em um registro central que esteja disponível (RECKZIEGEL, 2009).

3.3.2 **Consumidor de serviços**

É qualquer um que se utilize de um Web Service. Através de uma descrição disponibilizada pelo fornecedor de serviços, ele tem informação dos serviços e o caminho para o mesmo, por meio de uma pesquisa sobre o registro publicado (RECKZIEGEL, 2009).

3.3.3 ***Broker*** **dos serviços**

Seria a localização central onde o fornecedor de serviços estabelece seus Web Services, e no qual um consumidor de serviços pode pesquisá-los. Ou seja, é onde encontram informações como detalhes da empresa, inclusive detalhes técnicos, no registro do serviço. (RECKZIEGEL, 2009).

3.4. PADRÕES FUNDAMENTAIS DE *WEB SERVICES*

Existem cinco padrões fundamentais de *Web Services.* Dois deles são padrões gerais que existiam antes e foram utilizados para realizar a abordagem de Web Services (JOSSUTIS, 2008).
Esses padrões seriam camadas de comunicação entre aplicações de *Web Services,* elementos que encapsulam a requisição e a reposta entre um servidor e um cliente.
As camadas utilizadas são:

- HTTP (*Hypertext Transfer Protocol*)
- XML (*Extensible Markup Language*)
- *SOAP* (*Simple Object Access Protocol*)
- WSDL (*Web Services Definition Language*)
- UDDI (*Universal Discovery, Description and Integration*)

3.4.1 **HTTP**

O *Hypertext Transfer Protocol* é um protocolo de aplicação responsável pelo envio do *Web Service* pelas redes, utilizando tecnologias de Internet para o transporte de mensagens SOAP e documentos WSDL de um computador para o outro.

O seu funcionamento se baseia no tratamento de pedidos e respostas entre cliente e servidor na WWW (*World Wide Web)*. Ele passou a existir da necessidade de distribuir informações pela Internet e para que essa distribuição tornar-se viável foi necessário criar uma forma padronizada de comunicação entre os clientes e os servidores da *Web* e entendida por todos os computadores ligados à Internet. A partir desse momento, o protocolo HTTP passou a ser aplicado para a comunicação entre computadores na Internet e a apontar como seriam concretizadas as transações entre clientes e servidores, através do uso de regras básicas.

3.4.2 **XML**

O *Extensible Markup Language* (linguagem de marcação extensível) *é* um padrão desenvolvido em 1.996 pelo W3C (*World Wide Web Consortium*) que consisti em um grupo responsável pela definição de diversos padrões existentes na Internet (RECKZIEGEL, 2009).

Tem como característica principal seu formato de texto simples e flexível, permitindo estruturar e manipular seus dados, assim apresentando uma grande versatilidade na troca de dados entre aplicações comerciais e institucionais através da internet independente de plataforma e linguagem.

Um documento XML é organizado em forma de árvore, onde possui um elemento raiz, de onde se ramificarão os outros elementos. Essa estrutura em árvore estabelece a definição e o tratamento dos documentos XML, facilitando a busca de um determinado elemento ou grupo de elementos para serem processados (RUELA, 2008).

Na Figura 03, podemos observar um exemplo de documentação que apresenta os padrões XML.

```
...
<carros>

  <carro>
     <nome>Palio</nome>
     <marca>Fiat</marca>
     <ano>2010</ano>
   </carro>
  <carro>
     <nome>Fox</nome>
     <marca>VW</marca>
     <ano>2011</ano>
  </carro>
</carros>

....
```

Figura 03 - Exemplo de marcação XML.

As aplicações que utilizam *Web Services* indiretamente acabam se beneficiando do uso da tecnologia do XML para a troca e recepção das informações, o dado recebido pelo cliente, pode ser manuseado, alterado e visualizado sem a necessidade de reativar o servidor, proporcionando ao sistema um melhor rendimento dos seus processos e reduzindo o trafego de dados na comunicação entre cliente-servidor.

3.4.3 **SOAP**

O SOAP (*Simple Object Access Protocol*) é um protocolo baseado em XML utilizado para a troca de informações entre *Web Services,* no qual é possível definir o formato do cabeçalho e corpo de uma mensagem (JOSSUTIS, 2008).

Por ser código-aberto e ter sua especificação servindo de base para comunicação entre as aplicações, SOAP teve uma grande adoção pela maioria das grandes empresas de hardware e software. Sendo uma das principais camadas de comunicações do *Web Service,* embora não seja necessário o estudo do seu funcionamento para se criar e consumir um *Web Service*, é importante ter um entendimento geral sobre o protocolo, afim de lidar com eventuais falhas e problemas com a interoperabilidade entre as plataformas no uso de *Web Services*

A mensagem SOAP é um tipo de documento XML formado por um envelope composto de duas partes: cabeçalho (*header*) e o corpo da mensagem (*body*).

O envelope SOAP especificado pela *tag* <*Envelope*>, é o elemento raiz da mensagem que contém um cabeçalho opcional e um corpo obrigatório. O cabeçalho SOAP apontado pelo elemento <*Header*> pode ser usado para adicionar

características a uma mensagem SOAP. O corpo SOAP é o campo da mensagem SOAP que contém as informações que precisam estar constando dentro do elemento <*Body*> a serem trocadas entre aplicações através da mensagem.
A figura 04 ilustra os elementos da mensagem SOAP.

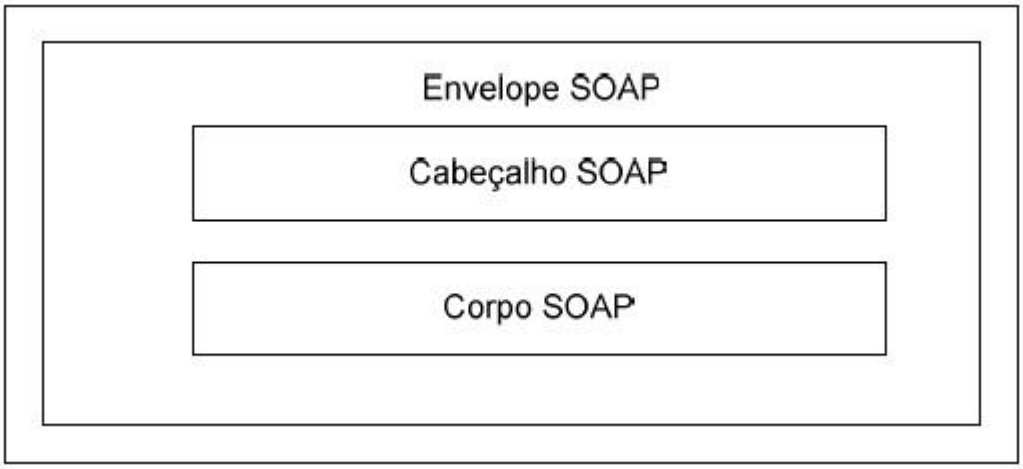

Figura 04 - Formato de mensagem SOAP.

3.4.4 **WSDL**

Web Services Description Language, o WSDL é a linguagem de descrição de um Web Services recomendada pela W3C, ela define uma padronização para a descrição dos serviços, externos ou interfaces que são proporcionados por uma aplicação especifica (RECKZIEGEL, 2009).

O objetivo principal do WSDL é descrever as interfaces expostas e marcar a localização dos seus serviços, em um local confiável na rede, o qual permite que o cliente acesse de maneira segura. Por ser um documento XML, sua leitura se torna simples e acessível. O WSDL faz a descrição de um serviço em duas partes, sendo que a primeira representa uma definição abstrata independente do protocolo de transporte de alto nível, enquanto a segunda faz uma representação da descrição da ligação específica para o transporte na rede Um documento WSDL determina o serviço como uma rede de coleções de parâmetros (RUELA, 2008).

3.4.5 **UDDI**

O *Universal Descripton, Discovery and Integration* (UDDI) é uma especificação técnica que tem como objetivo descrever, descobrir e integrar os *Web Services*, sendo um elemento central do grupo de padrões que compõe a pilha de componentes dos serviços Web (RUELA, 2008).

Antigamente, não existia padrão na Internet para as empresas localizarem seus clientes com informações sobre seus produtos e serviços e nem possuía um método de como integrar cada um dos seus sistemas e processos, a UDDI surgiu com o propósito de preencher essa lacuna, no qual tem por finalidade ser um mediador do serviço, possibilitando ao cliente requisitante encontrem um fornecedor do serviço apropriado.

Quando desenvolvemos um *Web Service*, necessitamos que os serviços oferecidos sejam acessados em algum lugar da Internet por um cliente. Uma das possibilidades é fazer com que a aplicação cliente conheça a URI do serviço.

A abordagem do UDDI baseia-se num registro distribuído de organizações e descrições dos respectivos serviços, implementado num formato XML comum. O componente principal do UDDI é o registro, que corresponde a um documento XML utilizado para descrever uma organização e os seus serviços (RUELA, 2008).

4. TECNOLOGIA GOOGLE ANDROID

O Android é um sistema operacional aberto criado pelo Google junto com a *Open Handset Alliance* (OHA) que é um grupo formado pelas mais importantes empresas do mercado em telefonia móvel, com o intuito de revolucionar o desenvolvimento de aplicações para dispositivos móveis, tanto para *smartphones* quanto a *tablets PC*.
É um sistema operacional fundamentado no *kernel* 2.6 do Linux, por ser multitarefa é responsável em gerenciar todas as tarefas e recursos de um dispositivo, possui uma vasta biblioteca de classes, contendo métodos para as mais variadas tarefas.

4.1. INTRODUÇÃO

Com o avanço da tecnologia, os aparelhos foram evoluindo e ganhando outra nomenclatura, o chamado *smartphone*, hoje muitos usuários modernos, estão procurando cada vez mais aparelhos com diversos recursos, como por exemplo: câmera de alta definição, jogos, acesso a internet, *bluetooth*, entre outras novidades. Em muitos países desenvolvidos já é possível utilizar o celular como uma espécie de cartão de crédito, como conferir extratos de conta-corrente de banco (LECHETA, 2010).
Dessa forma o uso de *smartphones* vem se tornando de grande utilidade e praticidade nos cumprimentos de tarefas do dia-a-dia tanto nestas organizações quanto para o uso particular.
Mas para que isso se torne realidade, é imprescindível uma plataforma bastante flexível e confiável, existem hoje vários tipos de soluções mobile, entre elas, as mais importantes são *Blackberry, Symbian, Palm OS, Iphone OS, Windows Mobile* e por último o noviço *Google Android*.
Para acompanhar o progresso dos *smartphones* existia uma necessidade em definir uma plataforma única, com código-aberto, flexível e moderna, nesse presságio a *Google* junto com a (OHA) que é um grupo formado pelas mais importantes empresas do mercado em telefonia móvel, criaram até então a grande responsável por revolucionar o desenvolvimento móvel, o *Google Android*, que versa em uma plataforma de desenvolvimento para aplicativos móveis, baseada no *kernel* 2.6 do Linux, por ser multitarefa, tem a autonomia de gerenciar a memória e os processos,

permitindo que diversas aplicações sejam executadas ao mesmo tempo, possui uma interface gráfica bastante rica e várias funcionalidades que agradam tanto ao usuário comum como o mercado corporativo (LECHETA, 2010).

Figura 05 - Logotipo Google Android.

4.2. SISTEMA OPERACIONAL LINUX

O sistema operacional (SO) do *Android* foi fundamentado no *kernel* 2.6 do Linux, é responsável por fornecer serviços do núcleo do sistema como gerenciamento de memória, processos, *threads*, serviços de segurança, além de redes e drivers.

Por ser multitarefa, permite que diversas aplicações e processos possam ser executados simultaneamente por tempo indeterminado, sem que o usuário perceba.

O *kernel* do SO por ser responsável em realizar todo o controle de memória tem autonomia em decidir finalizar algum processo a fim de liberar memórias e recursos, em alguns casos quando necessário pode restaurar o mesmo processo posteriormente quando a situação estiver sob controle (LECHETA, 2010).

A segurança do *Android* é baseada na segurança do Linux, utiliza do conceito de um ambiente de simulação, aonde a aplicação é executada em um único processo que por sua vez possui uma *thread* dedicada. Para cada aplicação instalada no celular é criado um usuário no sistema operacional para ter acesso a sua estrutura de diretórios, por padrão não tem permissões atribuídas a fim de negar o acesso de outros usuários a esta aplicação (LECHETA, 2010).

4.3. ARQUITETURA DO ANDROID

O *Android* é uma plataforma que abrange desde sistema operacional até o aplicativo, a sua arquitetura é dividida em várias camadas: *Applications, Application Framework, Libraries* e *Android Runtime;* e *Linux Kernel*, como é possível visualizar na Figura 06.

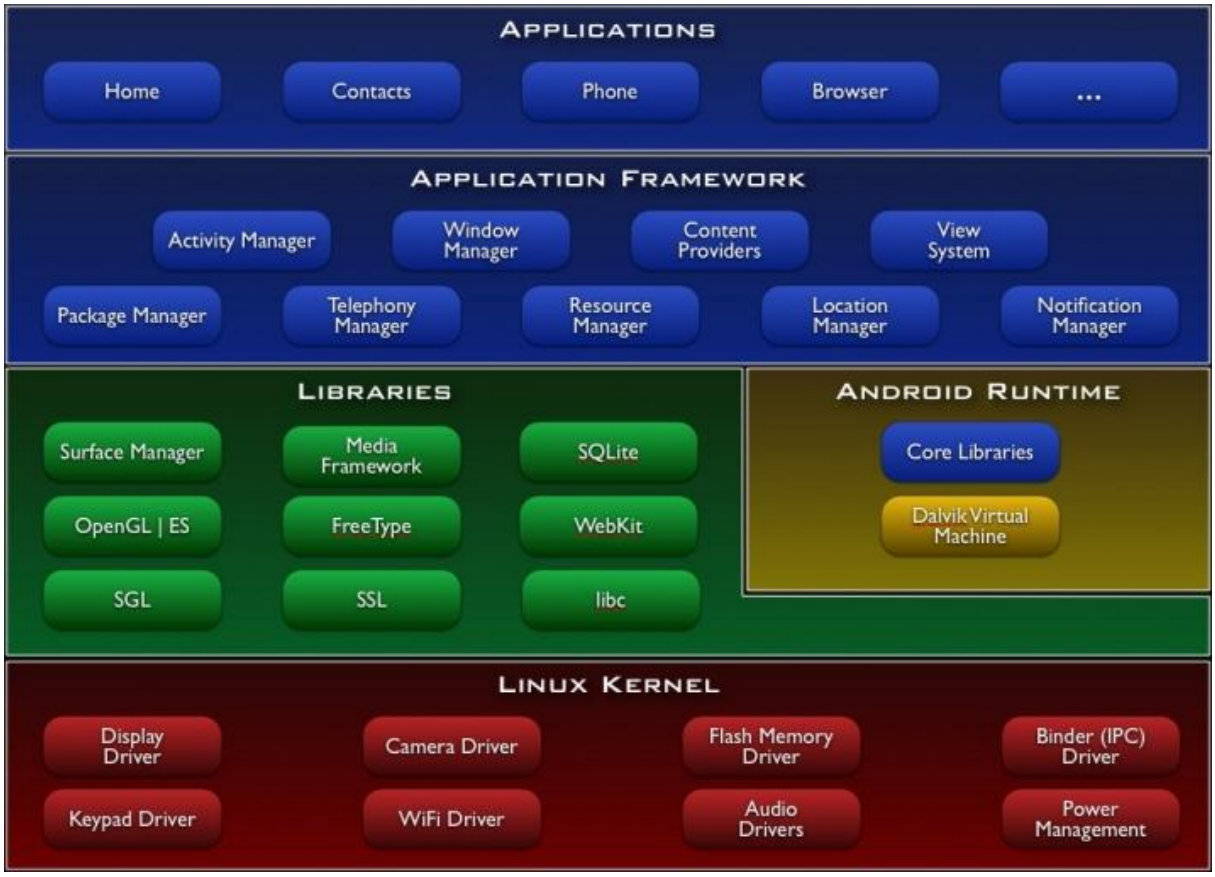

Figura 06 - Camadas da Arquitetura da plataforma *Android.*

4.3.1 **Camada *Applications***

Na camada *Applications*, encontrar-se localizada as aplicações que podem ser executadas sobre a plataforma, podem ser tanto aplicações nativas, que incluem um cliente de e-mail gerenciador de contatos, calendário, mapas, navegador *web*, programa de SMS, como as que serão criadas pelos os desenvolvedores (RABELLO, 2008).

4.3.2 **Camada *Application Framework***

Na camada *Application Framework*, estão todas as APIs que são utilizadas para o desenvolvimento da aplicação, como um rico e vasto conjunto de gráficos, provedores de conteúdo, que possibilita uma aplicação acessar dados de outras aplicações ou compartilhar seus próprios dados, como também outras atividades, por exemplo, gerenciadores de notificações, pacotes e recursos, no qual permite controlar os recursos previamente alocados, como a mudança entre os processos (RABELLO, 2008).

Pacote	Descrição
android.util	Contém várias classes utilitárias (classes de containers, utilitários XML)
android.os	Contém serviços referentes ao sistema operacional, passagem de parâmetros e comunicação entre processos.
android.graphics	Pacote principal dos recursos gráficos.
android.text **android.text.method** **android.text.style** **android.text.util**	Suporte para um conjunto de ferramentas de processamento de texto, suporte ao formato de texto rico (RTF), métodos de entradas, etc.
android.database	Contém APIs para comunicação com o banco de dados SQLite
android.content	APIs de acesso a dados no dispositivo, como as aplicações instaladas e seus recursos.
android.view	O pacote principal que contém os principais componentes de interface gráfica.
android.widget	Contém widgets prontos (botões, listas, gerenciadores de layout, etc) para serem utilizados nas aplicações
android.app	APIs de alto-nível referents ao modelo da aplicação. É implementada por meio de Activities (Atividades)
android.provider	Contém várias APIs para padrões de provedores de conteúdos (content providers)
android.telephony	APIs para interagir com funcionalidades de telefonia
android.webkit	Inclui várias APIs para conteúdos de context web, bem como um navegador embutido que pode ser utilizado por qualquer aplicação.

Tabela 01 - As principais APIs da plataforma *Android* (RABELLO, 2008).

4.3.3 **Camada *Libraries***

Na camada *Libraries* encontram-se às bibliotecas nativas escritas em C/C++, utilizadas pela plataforma *Android*. Estão contidas nessa camada a biblioteca C padrão (Libc), suporte a diversos formatos de multimídia, visualização de camadas 2D e 3D, funções para navegadores web, funções para gráficos 2D, funções de aceleração 3D via *hardware* e renderização 3D, conversões de fontes bitmap e vetor, por fim funções de acesso ao banco SQLite (RABELLO, 2008).

4.3.4 **Camada *Androd Runtime***

Um dos componentes desta camada são as core *libraries*, um conjunto de bibliotecas que fornece a maioria das funcionalidades disponíveis nas principais bibliotecas da linguagem de programação Java, outro componente é o *Dalvik Virtual Machine*, no qual consiste em uma máquina virtual, que tem a funcionalidade de dar suporte a execução de toda e qualquer aplicação, nessa camada observamos que ela dá condições para que as aplicações baseadas na plataforma sejam executadas (DEVELOPERS, 2011).

4.3.5 **Camada *Linux Kernel***

Na camada Linux *Kernel***,** está localizado o sistema operacional da plataforma, baseado no *kernel* 2.6 do Linux. Ela é responsável por fornecer serviços do núcleo do sistema como, como gerenciamento de memória e processos, pilhas de redes, segurança, etc (RABELLO, 2008).

4.4. *FRAMEWORK* DE DESENVOLVIMENTO

Framework de desenvolvimento constitui em uma estrutura que tem por finalidade auxiliar o desenvolvimento de aplicativos, podendo ser um conjunto de classes, ferramentas e recursos que automatizam e facilitam na escrita de códigos e testes (SILVA, 2010).

Para o desenvolvimento dos aplicativos *Android*, tem-se a disposição à linguagem Java, com as suas principais bibliotecas e conjuntos de componentes como gerenciadores de janela, recursos, telefone, atividade e o provedor de conteúdo. No ambiente de desenvolvimento empregado existem vários subsistemas, aplicativos e serviços, que auxiliam na criação dos aplicativos *Android*, como o emulador, o *debugger*, o *plug-in Android Development Tools* (ADT) para IDE Eclipse, dentre outros (DIMARZIO, 2008) (SILVA, 2010).

O desenvolvimento no Eclipse com o *plug-in* ADT é altamente indicado e a forma mais simples para iniciar na criação do aplicativo, auxilia na concepção dos códigos, uma vez que amplia os recursos e interface gráfica do Eclipse, também adiciona suporte integrado com o projeto e ferramentas Android, permitindo a integração de

ferramentas, editores XML construção de extensões, execução de depuração e execução do aplicativo, do modo a ser mais veloz e simples.

4.5. ANDROID SDK

Android SDK é o software que fornece as ferramentas e bibliotecas indispensáveis ao desenvolvimento de aplicações no *Android*, na sua composição temos ferramentas utilitárias, uma API completa para a linguagem Java e um emulador para simular o celular que pode ser executado como um aplicativo comum, embora exista um plug-in para a IDE Eclipse que proporciona exatamente integrar o ambiente de desenvolvimento java com o emulador (LECHETA, 2010).

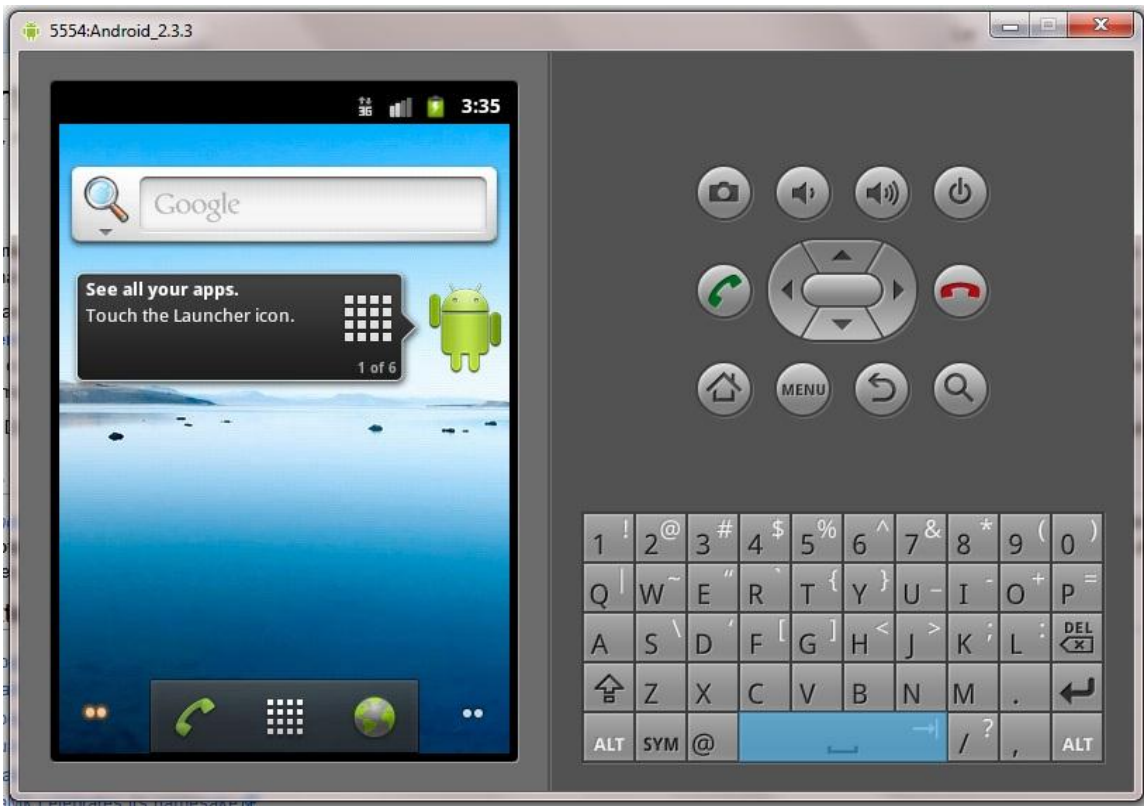

Figura 07 - Emulador do Android.

4.6. MÁQUINA VIRTUAL *DALVIK*

Dalvik é um processo de máquina virtual baseada em registradores, desenvolvida pela Google, onde foi projetada e escrita pelo Dan Bornstein junto com outros

engenheiros, tem por características solicitar pouca memória e permitir que vários aplicativos sejam executados ao mesmo tempo, depois de compilado o *bytecode* (.class) é convertido para o formato *.dex* (*Dalvik Executable*), que simboliza a aplicação do Android compilada.

Após a conversão, os arquivos *.dex* e os demais recursos utilizados pela aplicação são compactados apenas em um único arquivo com a extensão *.apk* (Android *Package File*), que representa a aplicação final, pronta para ser distribuída e instalada em qualquer dispositivo com Android (LECHETA, 2010).

4.7. GOOGLE MAPS

A *Google Maps* é um serviço gratuito de visualizações de mapas e imagens de satélites, fornecido pela empresa Google, disponível aos desenvolvedores utilizarem em seus sites desde que seu uso seja público.

O seu lançamento e sua disponibilidade deram por volta do ano 2.005, no qual caracteriza-se em um site com uma interface simples, afim de realizar pesquisas precisas com um ótimo desempenho, logo em seguida foi lançada a *Google Maps* API, que versa essencialmente de um conjunto de classes *JavaScript* que auxilia os desenvolvedores no acesso dos serviços disponibilizados pelo *Google Maps*.

É um dos recursos que mais chamam atenção na plataforma *Android*, pela sua grande facilidade em integrar sua API no desenvolvimento de aplicativos de localização com GPS, seja uma consulta por endereços, calculo de uma rota a ser percorrida de uma localidade para outra, movimentação de mapa, ou até mesmo função de zoom. Na Figura 08, é possível verificar a representação do fluxo de uma pesquisa no *Google Maps*.

Figura 08 – Esquema de uma solicitação de um mapa ao *Google Maps*.

5. DESENVOLVIMENTO DO APLICATIVO

Neste capitulo será apresentado à modelagem e a interface do trabalho proposto, aonde será divididos em partes para melhor entendimento das fases a ser executadas.

5.1. REDECA

Conhecido como Rede Eletrônica, o sistema (software) foi desenvolvido pela Fundação Telefônica - em parceria com oito municípios paulistas - e tem como intuito organizar, otimizar e aprimorar a qualidade do sistema de atendimento infanto-juvenil (REDECA, 2011).

O sistema facilita o trabalho em rede, integrando informações a respeito do atendimento e do desenvolvimento de crianças e adolescentes nos municípios. Isso permitirá um registro único de cada criança, centralizando informações relativas ao seu desenvolvimento, como saúde, educação e assistência social (REDECA, 2011).

A iniciativa envolveu, durante dois anos, articulação e ações de capacitação que alcançaram mais de 400 organizações governamentais e não governamentais nas cidades de Araçatuba, Bebedouro, Diadema, Guarujá, Itapecerica da Serra, Mogi das Cruzes, São Carlos e Várzea Paulista (REDECA, 2011).

O resultado obtido foi um software básico expansível e que contemplasse as necessidades dos oito municípios e provavelmente muitos outros. Os objetivos do grupo é dar continuidade no suporte, gestão de novas funcionalidades e melhoria contínua nas existentes (REDECA, 2011).

5.1.1 **Redeca Assis**

A Rede Ciranda da Criança e do Adolescente de Assis é um projeto subsidiado pela Fundação Telefônica que possui como alvitre a atuação integrada dos serviços de atendimento à criança e ao adolescente, contando com parcerias de diversas instituições para que suas atividades sejam realizadas (CIRANDA, 2011).

A Fundação Educacional do Município de Assis-FEMA é um dos principais parceiros do Projeto, contribuindo desde as ações iniciais elaboradas pelo Integraassis.

Tal parceria permitiu que todo o treinamento, implantação e acompanhamento do sistema REDECA fosse acompanhados por estagiários dos cursos de Ciências da Computação e Análise de Sistemas, sob orientação e supervisão e um coordenador responsável pelo processo (CIRANDA, 2011).

A UNESP- Campus de Assis foi outro forte parceiro, sobretudo por meio de núcleos de pesquisa, como o Núcleo de Estudos de Violências e Relações de Gênero-NEVIRG (CIRANDA, 2011).

5.2. DEFINIÇÃO DO PROBLEMA

O problema que será abordado nesse trabalho, consiste em desenvolver uma aplicação na plataforma *Android*, utilizando um módulo do sistema Redeca, a fim de ajudar e agilizar os processos de atendimentos das assistentes sociais.

Ele terá a funcionalidade de listar os atendimentos diários de cada assistente social, no qual o auxiliaria também no trajeto do seu ponto de origem até o destino, traçando uma rota a ser percorrida pelo aplicativo. No final de cada atendimento, seria enviado ao servidor, um formulário contendo um questionário com intuito de aprimorar e automatizar os serviços.

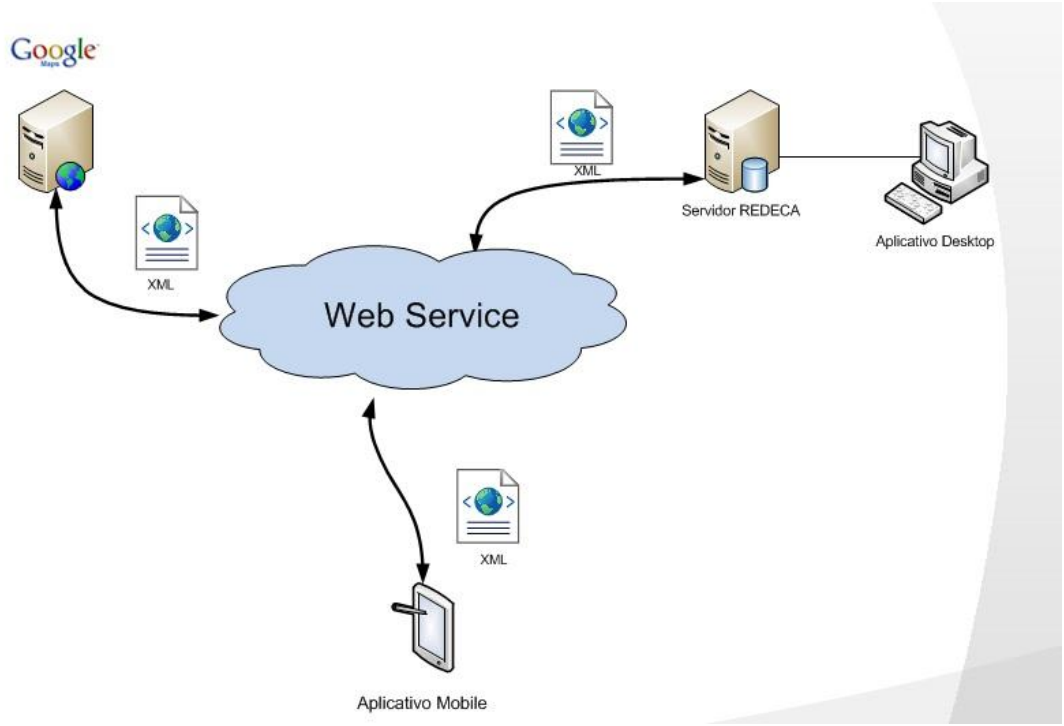

Figura 09 – Arquitetura do sistema.

O sistema terá com objetivo desenvolver as seguintes funcionalidades:

- Controle de acesso
- Lista entidades
- Listar assistentes sociais
- Listar atendimentos por assistente social
- Listar atendimento diário
- Apresentar dados de jovens assistidos
- Apresentar rota que deverá ser realizada pela assistente social

5.3. FERRAMENTAS UTILIZADAS NO DESENVOLVIMENTO DO APLICATIVO

As ferramentas utilizadas no desenvolvimento do aplicativo do trabalho serão: Eclipse (plataforma de programação Java e Android), Apache Tomcat/ Axis2.

5.3.1 **Eclipse**

O Eclipse é uma IDE open source desenvolvida pela IBM, utilizada para o desenvolvimento de aplicações Java, porém pode ser aproveitado em outras linguagens como Cobol, PHP, C++, etc
Possui um amplo suporte ao desenvolvedor a fim de atender as disparidades de diferentes programadores. Versão utilizada para o desenvolvimento do aplicativo será Helios Service Release 1.

5.3.2 **Java**

Linguagem de programação multiplataforma e orientada a objetos desenvolvida na década de 90 chefiada por James Gosling, na empresa Sun Microsystems. Java é uma linguagem compilada, diferente de outras linguagens de programação convencionais. O código fonte no processamento da compilação é transformado em um bytecode que em seguida é executado pela máquina virtual java, JVM.

5.3.2.1 **A Tecnologia Hibernate**

Framework desenvolvido em Java consolidado para fazer persistência dos dados em Java. Usa arquivos XML para facilitar a relação do mapeamento de atributos de uma

base de dados tradicional com os modelos de objetos de uma aplicação (mapeamento objeto-relacional).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
    "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
    "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="bean.Login" table="auth_user">
        <id name="Id">
            <generator class="native"></generator>
        </id>
        <property name="login"></property>
        <property name="pass"></property>
        <property name="name"></property>
    </class>
</hibernate-mapping>
```

Figura 10 – Mapeamento da classe no Hibernate.

O Hibernate possui a sua própria linguagem de SQL chamada HQL, que é convertida para SQLs características de cada banco de dados.
No desenvolvimento da aplicação foi utilizado Criteria Query API do Hibernate que permite construir query expression em Java para consultas no banco de dados. A API é uma alternativa as consultas HQL (Hibernate Query Language) e também ao SQL tradicional.

```
@Override
public List<T> login(String login, String pass) {
    Session session = HibernateUtil.getSessionFactory().getCurrentSession();
    session.beginTransaction();
    Criteria criteria = session.createCriteria(classe)
            .add(Restrictions.eq("login", login))
            .add(Restrictions.eq("pass", pass));

    List<T> lista = criteria.list();
    session.getTransaction().commit();
    return lista ;
}
```

Figura 11 – Código utilizando Criteria.

5.3.2.2 **Biblioteca JDOM**

JDOM é uma biblioteca open source desenvolvida em Java, modelo baseado em objeto de documento para XML, que foi projetado especificamente para a plataforma

Java para que ele possa tirar proveito de seus recursos de linguagem. JDOM integra com o Document Object Model (DOM) e Simple API for XML (SAX), suportes XPath e XSLT . Ele usa analisadores externos para criar documentos.

Atributos dessa biblioteca:

Document: Esta classe representa o documento inteiro, esta classe pode ter apenas um elemento que seria o elemento raiz do documento(elemento root/pai), comentários e outros elementos de definição.

Element: Esta classe representa um elemento do documento XML, são os elementos que compõem o documento, ele pode conter outros elementos chamados de filhos, atributos, valores e comentários.

Attribute: Esta classe representa um atributo de um elemento XML.

5.3.3 **Apache Tomcat/ Axis2**

O Apache Tomcat é um servidor de aplicação desenvolvido para o padrão JEE. Suporta tanto servlets como jsp, outra funcionalidade é a sua utilização como servidor web provendo serviços de HTTP Java. Possui ferramentas para configurações e gerenciamento, podendo ser feito também utilizando XML.

5.4. DIAGRAMA DE CASO DE USO

5.4.1 **Diagramas de Casos de uso geral do sistema**

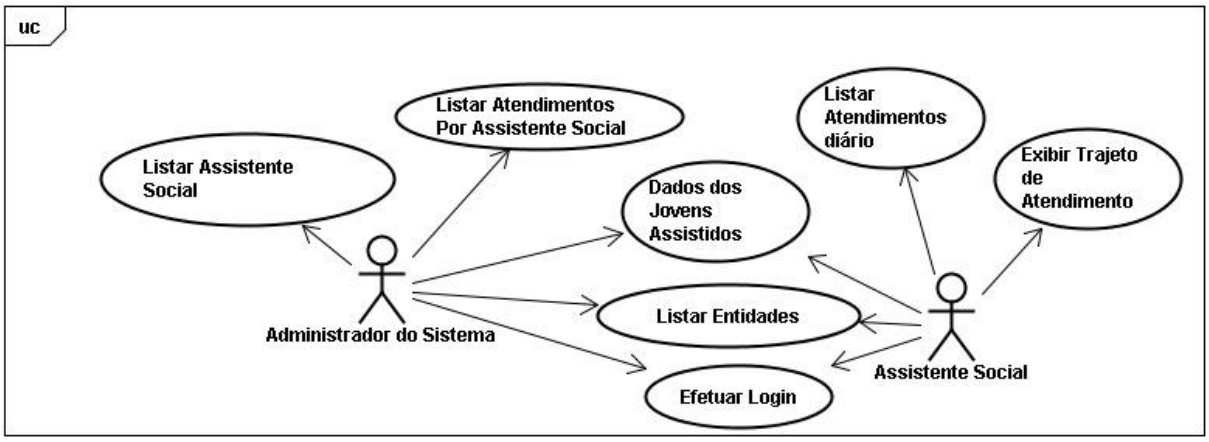

Figura 12 - Casos de Uso Geral do Sistema.

5.4.1.1 **Administrador do Sistema**

Seu acesso é efetuado por login, tem privilégios de manipular todas as informações do sistema.

5.4.1.2 **Assistente Social**

Seu acesso ao sistema é obtido através de login, tem por finalidade a consulta dos Casos: Dados dos Jovens Assistidos, Listar Entidades, Listar Atendimentos diários e Exibir Trajeto de Atendimento.

5.5. DIAGRAMA DE ENTIDADE RELACIONAMENTO (DER).

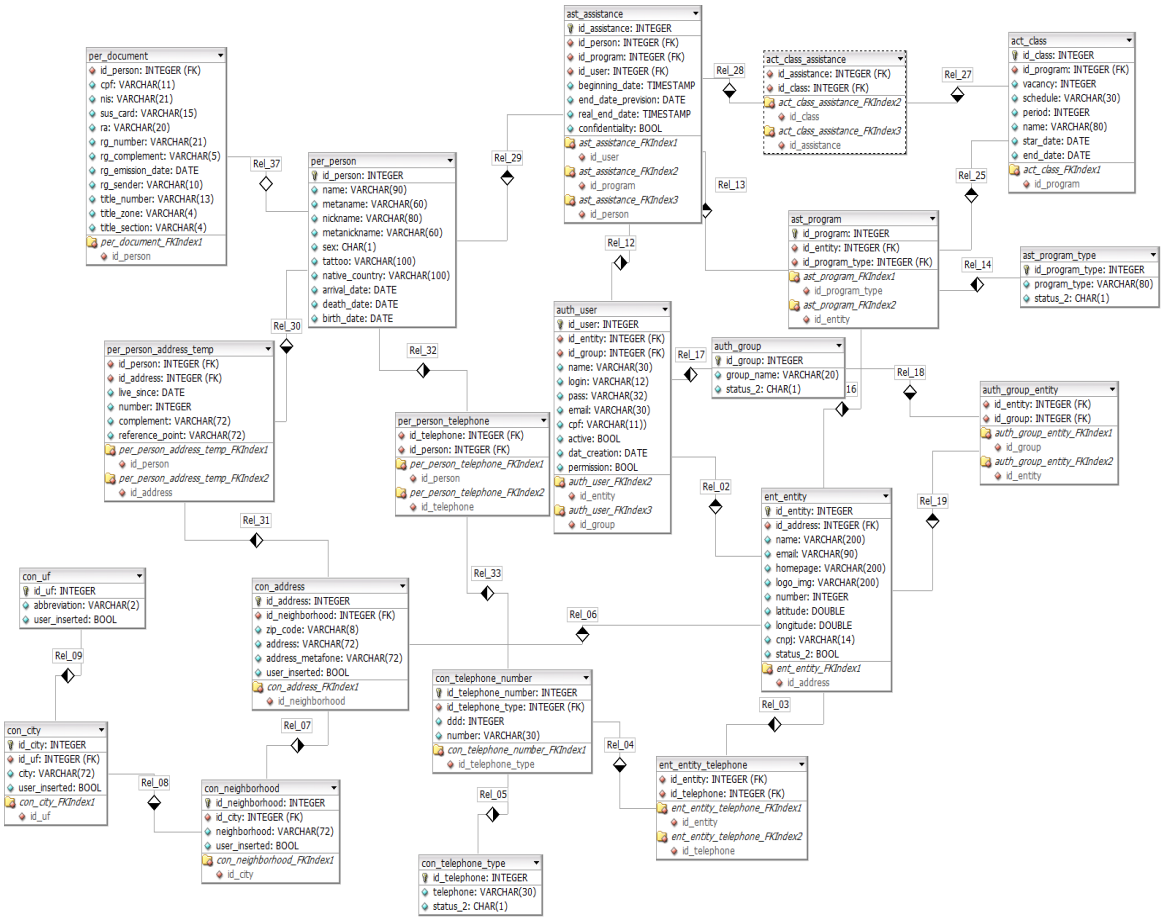

Figura 13 – Diagrama de Entidade e Relacionamento (REDECA).

5.6. APLICATIVO MOBILE.

A aplicação Mobile foi implementada utilizando a plataforma Google Android, comunicando-se com o servidor de aplicação desenvolvida nesse trabalho.
A figura 14 apresenta a tela principal do aplicativo, onde o usuário será autenticado no sistema.

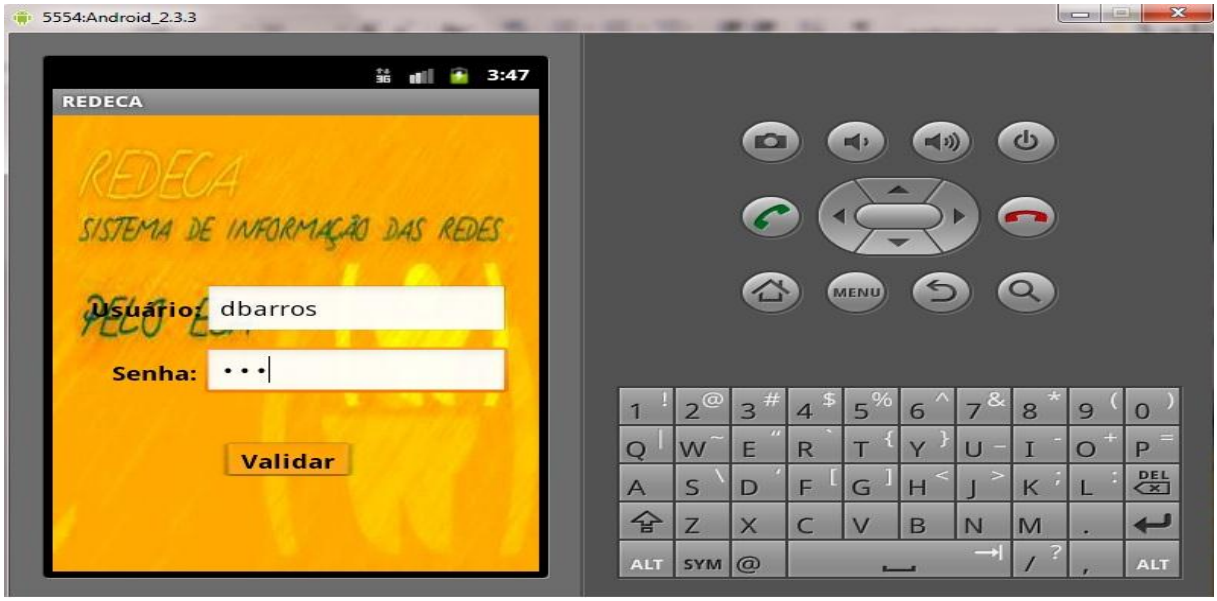

Figura 14 – Interface de Acesso Módulo Móvel REDECA.

Na figura 15 - Item a) escolhendo a opção Lista de Assistentes Sociais do menu principal, será exibida uma interface com a listagem de Assistentes Sociais conforme o item b.

Figura 15 – a) Menu b) Listagem de Assistentes Sociais (REDECA).

Na figura 16 - Item a) exibe o menu, com opções de Agendamento, Lista de Assistentes Sociais, Lista de Entidades, Busca por Contato, e fechar. No item b), escolhendo a opção Agendamento, invocará um sub-menu no qual o usuário terá a opção de escolher entre Atendimento diário ou Atendimento por Assistente Social.

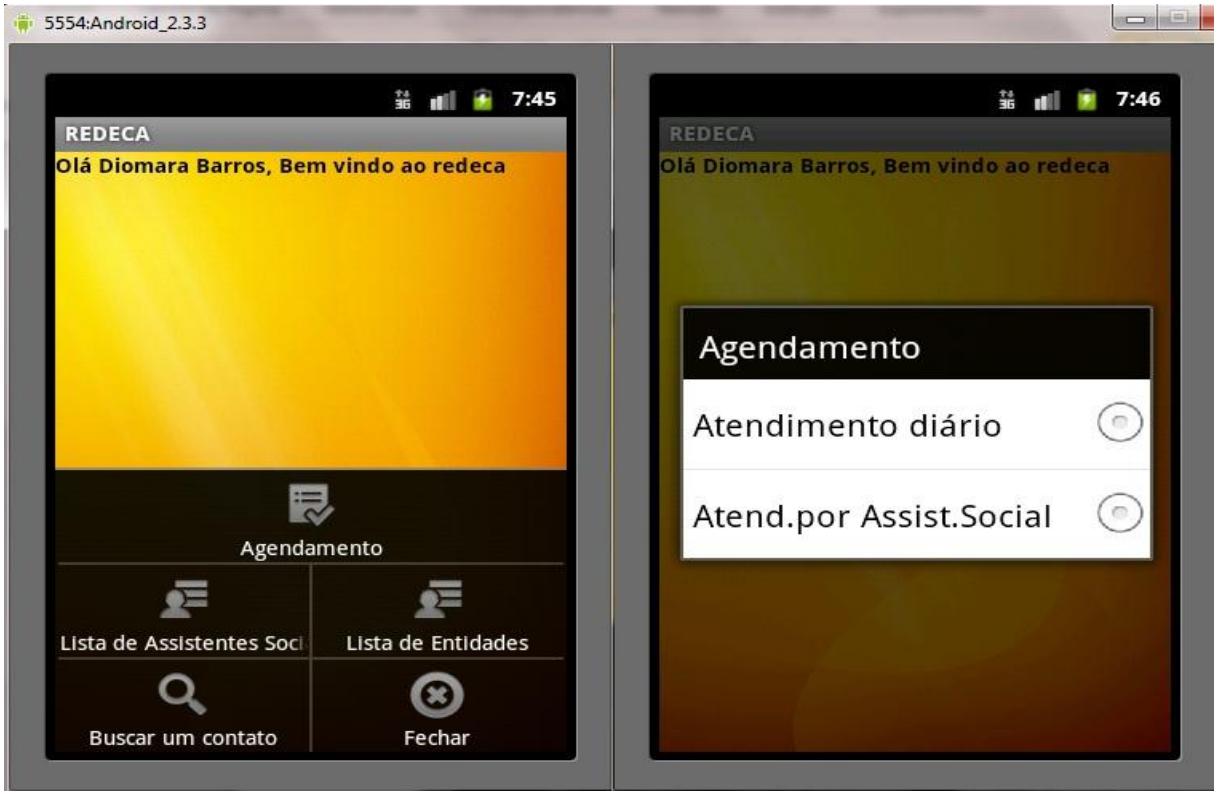

Figura 16 – a) Menu b) Agendamento (REDECA).

Figura 17 – a) Atendimento Diário b) Atend. por Asistente Social (REDECA).

Selecionando a opção Lista de Entidades do menu, uma tela será exibida com a listagem das entidades cadastradas no sistema, conforme a figura 18.

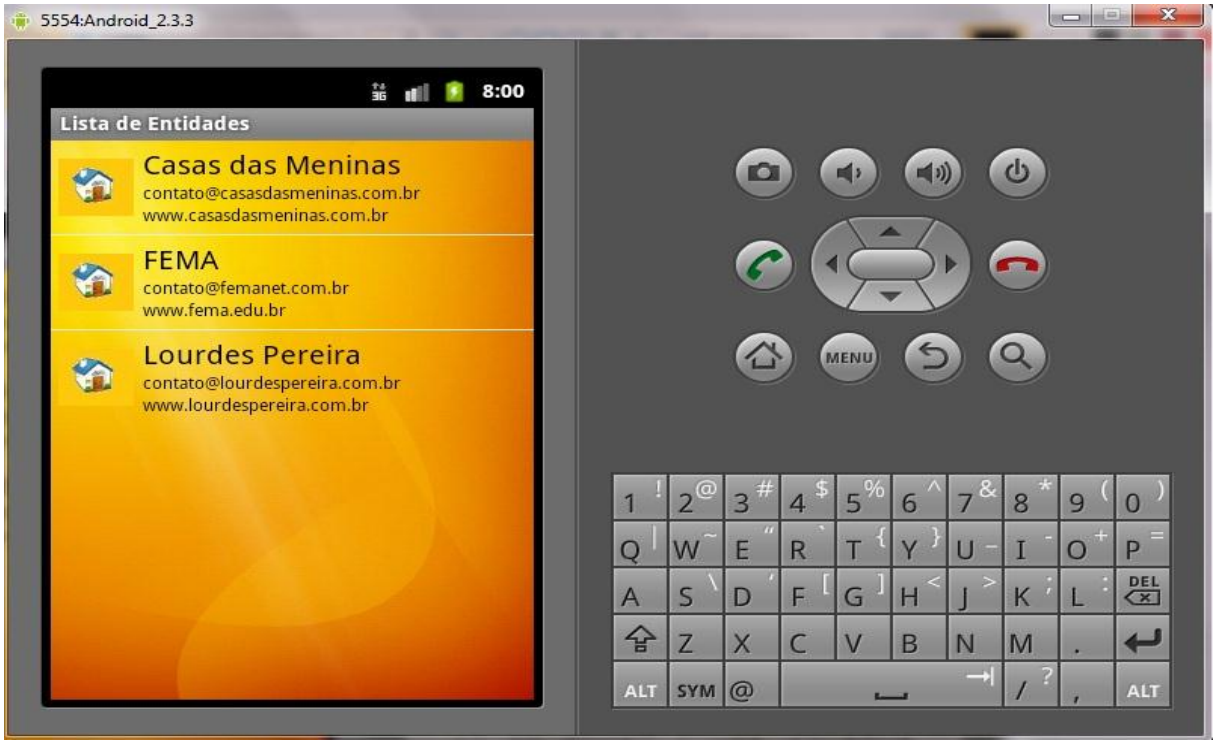

Figura 18 – Listagem de Entidades (REDECA).

Optando pela opção Busca Contato do menu, uma tela será exibida aonde o usuário efetuará a pesquisa dos jovens cadastrados no sistema, conforme a figura 19.

Figura 19 – Busca contato (REDECA).

6. CONCLUSÃO

Esse trabalho de conclusão de curso foi motivado pelo grande avanço tecnológico na área mobile e pela necessidade de mão-de-obra em que o mercado se encontra.

Foi necessário um estudo de diversas tecnologias dispostas no mercado, contudo foi escolhida a linguagem Java e Android na qual eram as mais adequadas para o desenvolvimento do trabalho proposto.

Algumas dificuldades surgiram em consequência do tema abordado ser novo e ainda pouco utilizado nos meios acadêmicos, o maior desafio consistiu se em aprender e entender os conceitos de *Web Services* e sua implementação, tanto na parte da construção do serviço como na utilização do *client* no aplicativo mobile.

Com o desenvolvimento Mobile, foi possível obter um aplicativo empregando todo o conhecimento adquirido na pesquisa, no qual será utilizado na apresentação final deste trabalho acadêmico e posteriormente em futuras consultas sobre o tema em meios acadêmicos.

REFERÊNCIAS

BARBOSA, Daniela. **Há 20 anos, celular só fazia ligação.** Economias e Empresas IG: Disponível em: <http://economia.ig.com.br/empresas/comercioservicos /ha+20+ anos+celular+so+fazia+ligacao/n1237899643507.html /> Acessado em: 01 mar. 2011.

CIRANDA, Rede. **Ciranda.** Rede Ciranda da Criança e Adolescente de Assis. Ed.Especial, Ano 01 nº01, fev. 2011.

DEVELOPERS, Android. **What is Android?**. Dev Guide: Disponivel em: <http://developer.android.com/guide/basics/what-is-android.html>. Acessado em 09 jun. 2011.

DIMARZIO,J.F., **Android-A Programmer's Guide**, McGraw Hill, 2008.

ERL, Thomas. **Services-Oriented Architeture: Concepts, Technology, and Design.** UpperSaddle River, NJ: Prentice Hall, 2005.

ESTADÃO, Agência. **Brasil fecha fevereiro com mais de 207,5 milhões de celulares.** Economias e Empresas IG: Disponível em: <http://economia.ig.com.br/ empresas/comercioservicos/brasil+fecha+fevereiro+com+2075+milhoes+ de+ celulares/n1300008310786.html/>. Acessado em: 28 mar. 2011.

JOSUTTIS, N. M. **SOA na Prática** – A Arte da Modelagem de Sistemas Distribuidos. Alta Books, 2008.

LECHETA, Ricardo, **Google Android 2º Edição**, Editora Novatec, 2010.

MEIER, R, **Professional Android Application Development**, Wrox, 2008.

MICROSOFT. **Perguntas Frequentes sobre SOA e BPM**. SOA e Processos de Negócios. Disponivel em: <http://www.microsoft.com/brasil/servidores/soa/about/faq .aspx >. Acessado em: 12 jun. 2011.

RABELLO, R. R., **Android: Um novo paradigma de desenvolvimento móvel.** Web Mobile Ed.18, p.06-12, 2008.

RABELO, R. J. **Arquiteturas Orientadas a Serviços. Disciplina Integração de Sistemas Corporativos**. Departamento de Automação e Sistemas. Universidade Federal de Santa Catarina. 2006.

RECKZIEGEL, Mauricio. **Gerenciamento de Infra-Estrutura de Medição usando Web Services.** Universidade Regional do Noroeste do Estado do Rio Grande do Sul, 2009.

REDECA. **Sistema de Informação das Redes pelo Eca**. Disponivel em: <http://redeca.sourceforge.net/>. Acessado em: 12 jun. 2011.

RUELA, Pedro Henrique de Oliveira. **Solução Web Services.** Faculdade de Jaguariúna, 2008.

SILVA, L.E.P. **Uma nova abordagem para Cálculo de Balanço Hidríco Climatológico.** Revista Brasileira de Computação Aplicada (ISSN 2176-6649), Passo Fundo, v. 2, n. 1, p. 2-16, mar. 2010.

TOSIN, Carlos. **Conhecendo o Android**. Dicas-L: Disponivel em: <http://www.dicas-l.com.br/arquivo/conhecendo_o_android.php>. Acessado em 09 jun. 2011.