ARQUITETURAS DE REDES DE RELACIONAMENTOS UTILIZANDO TECNOLOGIAS MÓVEIS

Daniel Gasparin[1], Dione Jonathan Ferrari[1]

[1]Faculdade de Informática de Presidente Prudente (FIPP) – Universidade do Oeste Paulista (UNOESTE). Rua José Bongiovani nº 700 – 19.050-920 – Presidente Prudente – SP – Brasil. dang@unoeste.edu.br, dferrari@unoeste.br

RESUMO

Redes sociais são serviços online que se concentram na construção e reflexão de relações sociais entre as pessoas pela internet, vários dispositivos poderiam ser inseridos a essas redes para aumentar a interação de seus usuários utilizando novas tecnologias como dispositivos móveis e GPS. O presente artigo propõe e apresenta um modelo de arquitetura com o objetivo de incluir dispositivos móveis, com modulo de GPS acoplado, as redes hoje conhecidas. A proposta integra as tecnologias J2ME, GPS e internet em seu desenvolvimento a fim de realizar seus devidos testes.
Palavras-chave: redes sociais;dispositivos móveis;gps;j2me

ARCHITECTURES OF NETWORKS OF RELATIONSHIPS USING MOBILE TECHNOLOGIES

ABSTRACT

Social networks are online services that focus on construction and reflection of social-political interactions among people over the Internet, multiple devices could be inserted to those networks to increase interaction from users using new technologies like mobile devices, and GPS. This paper proposes and presents an architectural model in order to include mobile devices, coupled with GPS module, the network now known. The proposal includes the J2ME technology, GPS and internet in their development to achieve their proper tests.
social network;mobile devices;gps;j2me
Keywords: social network;mobile devices;gps;j2me

1. INTRODUÇÃO

Este trabalho tem como proposta apresentar uma arquitetura da utilização de dispositivos móveis com modulo GPS acoplado as já existentes redes sociais. Para viabilizar o processo foi proposto o desenvolvimento de um protótipo com as tecnologias J2ME, módulos Java para Web, a API Google Maps e a API Open Social para o desenvolvimento de um LBS.

O LBS (acrônimo do inglês Location-based service) pode ser definido como serviços baseados em localização, a utilização deste serviço será dada por um dispositivo com acesso a internet que irá receber sinais dos satélites de GPS e se comunicar com uma base de dados recebendo e enviando informações baseadas em sua localização.

Para um melhor entendimento do assunto, este artigo foi dividido em 4 partes, na segunda parte será discutido a visão geral do trabalho na qual a estrutura básica do projeto será apresentada. Na terceira parte serão vistas as tecnologias utilizadas para a construção do protótipo e na quarta parte apresenta-se a conclusão deste trabalho juntamente com sugestões para futuros trabalhos.

2. VISÃO GERAL DO PROJETO

A intenção do projeto é apresentar um modelo de arquitetura que possibilite a comunicação de usuários das redes sociais utilizando tecnologias moveis como internet de banda larga para celulares e o GPS.

Nesta arquitetura, o dispositivo móvel enviará informações sobre o posicionamento do usuário, como latitude e longitude, para um servidor que irá guardar esses dados podendo disponibilizá-los para seus contatos (rede de amigos). O servidor web por usa vez tem o objetivo de processar os pedidos de *login*, gerenciar os usuários e seus contatos e exibir, através de um browser, um mapa utilizando o *Google Maps* e uma marcação contendo a localização de cada contato a ser consultado, veja na Figura 1 a visão geral do funcionamento do projeto.

Para melhor entendimento da Figura 1 o modelo foi dividido em duas partes: a primeira contém a parte do dispositivo móvel vendo seus processamentos e suas comunicações; e a segunda define as interfaces web, juntamente com o banco de dados.

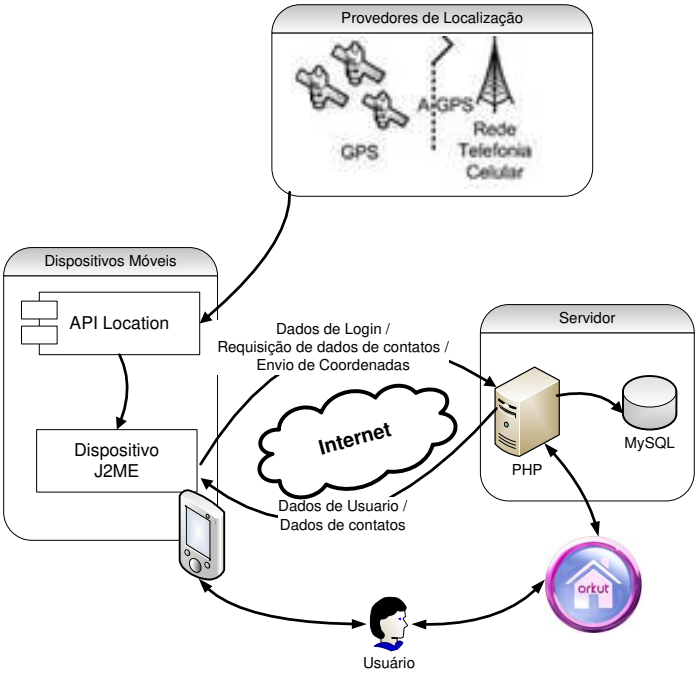

Figura 1: Visão geral do projeto.

2.1 Dispositivo Móvel

Os dispositivos móveis nesse projeto podem se comunicar com um serviço web para enviar e receber informações, e é nele que o LBS começa, pois pelo dispositivo móvel são coletadas as coordenadas, que é o principal propósito do projeto.

Nos dispositivos móveis temos algumas funcionalidades como o *login* de usuário, a captura e envio de coordenadas e o recebimento de dados de contatos com uma distância pré-definida.

Na Figura 2 apresenta-se a comunicação

do dispositivo móvel, recebendo dados GPS dos satélites e trocando informações com o servidor.

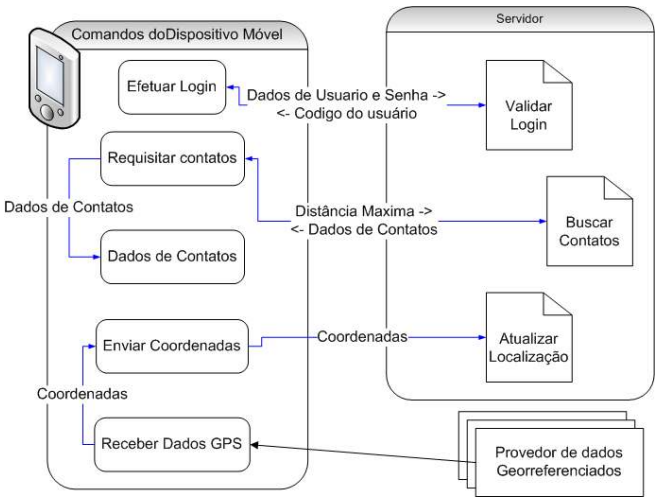

Figura 2: Comunicações do dispositivo móvel com satélites e com o servidor.

2.1.1 *Login* de Usuários

Em uma aplicação multiusuário, uma validação de nome de usuário e senha é indispensável, para garantir a segurança e a integridade das informações que serão coletadas e transmitidas, e também para garantir a segurança de todos os usuários que estarão transmitindo suas informações.

No login um usuário previamente cadastrado envia seu nome de usuário e senha para o servidor e lá é feita a verificação, devolvendo uma informação ao dispositivo dizendo se os dados conferem ou não. A Figura 3 apresenta a tela de *login* do protótipo deste projeto.



Figura 3: Tela de login do protótipo no dispositivo móvel.

2.1.2 Captura e envio de coordenadas

Para um serviço baseado em localização possa ser implementado deve haver serviços que recebam e transmitam a localização do usuário.

No LBS tudo tem início no dispositivo móvel que será responsável por coletar e enviar informações para o servidor serão gravadas a latitude, a longitude, a altitude e a data, juntamente com o código do usuário possibilitando assim a recuperação dos dados por terceiros ou pelo próprio usuário.

A Figura 4 Corresponde a tela de captura de coordenadas do protótipo.



Figura 4: Tela de captura e envio de coordenadas.

2.1.3 Recuperação de Contatos

Para a recuperação de contatos o usuário insere uma distância máxima em metros e de acordo com essa distância o servidor irá calcular quem está nesse perímetro e que na lista de contatos do usuário.

Para tal, o usuário deverá informar a distância máxima em metros e o servidor retornará dados dos usuários próximos como nome, e-mail e número de telefone para que o usuário além de receber os dados possa interagir de forma mais direta com seus contatos.

2.2 Módulo Web

O módulo web é responsável pelas requisições dos dispositivos móveis, nele são feitos o envio e recebimento de informações como informações sobre o usuário, suas informações geográficas e informações sobre seus contatos.

O servidor é composto por dois serviços: os módulos para comunicação com o servidor e as páginas para interação com o usuário.

A Figura 5 apresenta as páginas que fazem parte da construção do servidor e cada uma delas será explicada juntamente com sua função.

Na mesma figura podem-se observar as páginas sem suas devidas extensões, pois por se tratar de um modelo pode ser desenvolvida em diversas linguagens web.

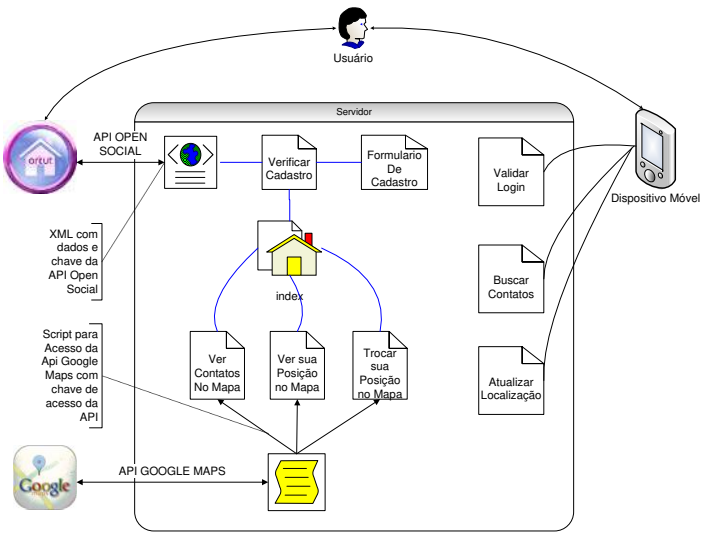

Figura 5: Arquitetura do servidor.

2.2.1 Serviços de comunicação com dispositivo móvel

Esse conjunto de páginas é responsável pela comunicação com o dispositivo móvel e irão receber e enviar informações. As paginas deste serviço podem ser vista ligadas com o dispositivo móvel na figura 5.

As solicitações do dispositivo móvel são enviadas via url pelo método get e irá receber os dados da pagina web após o processamento.

As paginas de serviço de comunicação com o dispositivo móvel são:

- Pagina de Validar Login:
  - Neste serviço, serão enviados o nome de usuário e senha e o sistema irá procurar no banco de dados, conferindo a autenticidade das informações, e retornar o código do usuário ao dispositivo móvel.
- Pagina de Buscar Contatos:
  - Nesta página uma solicitação será enviada ao servidor pelo dispositivo móvel para que todos os contatos do usuário dentro de uma distância escolhida sejam retornados

para o dispositivo móvel. Para isso, a página será solicitada passando o código do usuário e a distância pretendida em metros.

- Pagina de Atualizar Localização:
  - Esta página tem como função atualizar as coordenadas do usuário. Após a coleta de dados GPS o dispositivo móvel irá enviar essas coordenadas a esta página que por sua vez irá salva-los juntamente com a data e hora da coleta no banco de dados.

2.2.2 Serviços de interação com o usuário

As páginas de interação são as páginas que podem ser vistas pelos usuários. Possuem funcionalidades bem definidas e se diferem das páginas de comunicação com dispositivo móvel por terem um design gráfico ausente nos módulos de comunicação que só contam com o código propriamente dito.

Podem-se ver essas páginas na Figura 5, comunicando-se com o usuário através de uma rede social, neste caso o Orkut.

As paginas de serviço de interação com o usuário são:

- Pagina de Verificar Cadastro:
  - Toda vez que o usuário entrar no sistema por uma aplicação de uma rede social, ele será direcionado para a página de verificação de usuário, o código do usuário será fornecido pela rede social através da API Open Social, caso o usuário não estiver cadastrado ele será enviado para uma pagina de cadastro, caso contrario irá ser direcionado a pagina principal.
- Pagina de Formulário para Cadastro:
  - Na pagina de cadastro será requisitado do usuário o seu nome, e-mail, número de telefone celular, juntamente com um nome de usuário e senha que será utilizado para fazer o login no dispositivo móvel.
- Pagina de Principal (Index):
  - Na pagina principal se encontrará dados sobre a utilização do sistema, e uma opção para download da aplicação móvel para ser instalada no celular.
- Pagina de Visualização de contatos no mapa:
  - Nessa página será exibida uma relação de contatos em um mapa contendo a localização de cada contato exibindo um ponto vermelho para cada um juntamente com a localização atual do usuário sendo identificado como um ponto azul, o usuário poderá ver separadamente cada contato selecionando o contato pretendido, ao selecionar um contato será exibida somente a posição juntamente com as informações do contato como nome, e-mail e número celular.
- Pagina de Visualização da posição do usuário no mapa:
  - Nesta pagina é exibido um mapa com a sua posição atual, será disponibilizados os dados do usuário com a opção de alterá-los.
- Pagina de Atualizar posição do usuário

no mapa:

- Nesta pagina é exibido um mapa com a sua posição no qual ao clicar em qualquer área em branco será ativada uma função que irá perguntar ao usuário se deseja trocar sua posição, se a resposta for afirmativa serão gravados os novos dados sobre a localização do usuário como latitude, longitude e data da atualização.

2.2.3 Banco de dados

O banco de dados é responsável por guardar dados dos usuários, suas localizações e também a relação de todos seus contatos, na figura 6 vemos as tabelas do banco, juntamente com seus campos, repetindo que não foi utilizado os tipos de dados pois na construção de um protótipo poderá ser utilizada qualquer banco de dados que atenda os requisitos do projeto.

Veja na figura 6 o banco de dados do projeto.

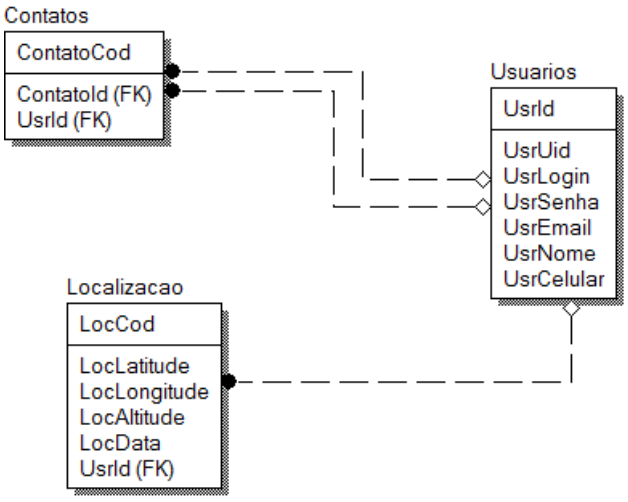

Figura 6: Banco de dados.

3. TECNOLOGIAS UTILIZADAS

Esta seção contem as principais tecnologias utilizadas no desenvolvimento do protótipo. Para testar a viabilidade do projeto foram escolhidas algumas linguagens que no futuro poderão ser substituídas por outras sem nenhum problema desde que seja seguido o modelo de arquitetura.

2.3 J2ME

Java Plataform Micro Edition, Java ME, ou simplesmente J2ME, é uma tecnologia que possibilita o desenvolvimento de softwares para sistemas embarcados e compactos como celulares, PDAs, *pagers, smartphones, palmtops*, e outra gama de dispositivos.

Esta tecnologia Java é focada na portabilidade de aplicativos, tanto que para o J2ME foi desenvolvida uma própria maquina virtual a KVM que está vinculada aos recursos limitados dos aparelhos e é responsável por atender as configurações do CLDC (*Conected Limited Device Configuration*) e do CDC (*Connected Device Configuration*), e cada recurso instalado devem ser atendidos por classes especialmente desenvolvidas para eles de acordo com o perfil MIDP (*Mobile Information Device Profile*).

A Figura 7 apresenta o funcionamento do J2ME dentro de um aparelho celular.

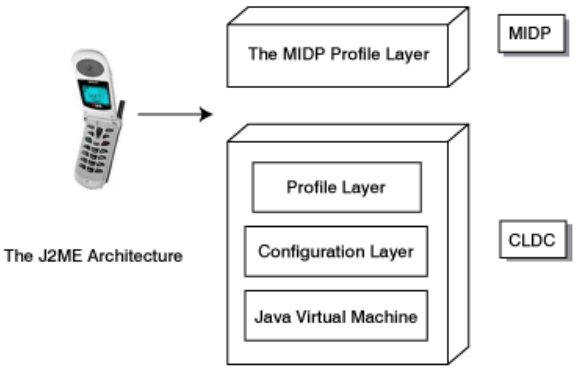

Figura 7: Arquitetura do J2ME.[1]

2.3.1 API Location (JSR 179)

Esta API traz um conjunto de classes, métodos e interfaces que permite que as aplicações Java ME possam ser sistemas LBS, veja a figura.

A JSR 179 trabalha com localização dinâmica, e não faz distinção ao método de busca da localização, pode ser um método como o GPS, mas também, pode ser um Cell ID (baseado nas Estações Rádio Base das operadoras de telefonia celular).

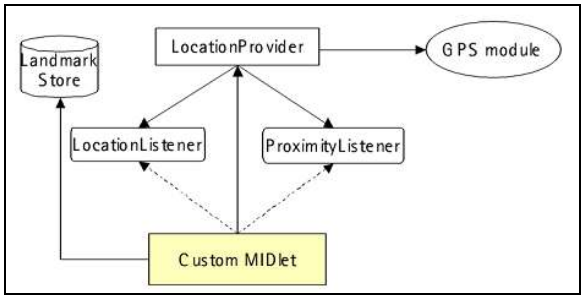

Figura 8: MIDlet usando a JSR 179 API Location.[3]

Outra questão importante sobre a API, é o fato dela funcionar corretamente somente em dispositivos com a CLDC 1.1, devido a necessidade do uso de números em pontos flutuante.

A Figura 9 apresenta um trecho de código do protótipo utilizando a *api location* para recuperar dados como latitude, longitude e altitude.

```
public void checkLocation() throws Exception
{
   String string;
   Location l;
   LocationProvider lp;
   Coordinates c;
   Criteria cr= new Criteria();
     cr.setHorizontalAccuracy(500);
     cr.setVerticalAccuracy(500);
     cr.setPreferredPowerConsumption
              (Criteria.POWER_USAGE_HIGH);
   lp= LocationProvider.getInstance(null);
   l = lp.getLocation(60);
   c = l.getQualifiedCoordinates();
   if(c != null ) {
      double lat = c.getLatitude();
      double lon = c.getLongitude();
      double alt = c.getAltitude();
      string = "\nLat: " + lat + "\nLong: "
               + lon + "\nAlt: " + alt;
    } else {
      string ="Location API failed";
    }
    midlet.displayString(c);
 }
```

Figura 9: Método de coleta de dados GPS.

2.3.2 Protocolo HTTP

Tanto na comunicação do dispositivo móvel com o servidor quanto do servidor com o dispositivo foi utilizado o protocolo HTTP por seu desempenho e simplicidade.

Para a conexão do HTTP utilizamos a classe connector que também serve para outros tipos de conexões, como arquivos, datagramas, etc. Esta classe procura a melhor classe para executar determinado protocolo, utilizando de uma maneira melhor a GCF (Generic Framework Connection, estrutura de conexão genérica) que aproveita as classes de conexões existentes no JSE adaptando-as para dispositivos móveis.

Na figura 10 podemos ver a hierarquia da GFC onde começa a classe connection, todos os tipos de conexões possíveis e onde esta localizado o protocolo HTTP dentro da GFC.

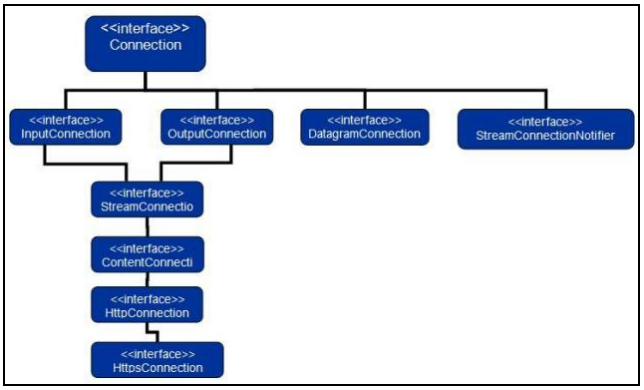

Figura 10: Hierarquia da GFC.

Após aberta a conexão HTTP com a classe conector podemos escolher 3 formas de enviar os dados, que são:

Get - solicita dados, que são enviados como parte da URL;

Post – solicita dados, que são enviados em um fluxo separado da URL;

Head – solicita meta-informações sobre determinado recurso.

Para a implementação do protótipo deve ser utilizada somente Get ou Post, veja na figura 11 e 12 a implementação do Get e Post.

```
// Data is passed at the end of url for GET
String url =
  "http://www.mycgiserver.com/servlet/corej2me.GetNpostServlet" +
   "?" +
   "account=" + tfAcct.getString() + "&" +
   "password=" + tfPwd.getString();
try
{
  http = (HttpConnection) Connector.open(url);
//----------------
// Client Request
//----------------
// 1) Send request method
http.setRequestMethod(HttpConnection.GET);
// 2) Send header information - none
// 3) Send body/data -  data is at the end of URL
//----------------
// Server Response
//----------------
iStrm = http.openInputStream();
// Three steps are processed in this method call
ret = processServerResponse(http, iStrm);
}
```

Figura 11: requisição por método GET

```
// Data is passed as a separate stream for POST (below)
String url =
    "http://www.mycgiserver.com/servlet/corej2me.GetNpostServlet";

try

{
  http = (HttpConnection) Connector.open(url);
  oStrm = http.openOutputStream();

  //----------------
  // Client Request
  //----------------
  // 1) Send request type
  http.setRequestMethod(HttpConnection.POST);

  // 2) Send header information. Required for POST to work
  http.setRequestProperty("Content-Type",
        "application/x-www-form-urlencoded");
  // 3) Send data
  // Write account number
  byte data[] = ("account=" + tfAcct.getString()).getBytes();
  oStrm.write(data);
```

Figura 12: requisição por método POST

2.3.3 LWUIT (Lightweight User Interface Tools)

LWUIT (Lightweight UI Toolkit) é uma biblioteca que permite criar interfaces gráficas do usuário (GUI) bastante atraentes, para dispositivos móveis ou quaisquer outros dispositivos que suportem o perfil MIDP, do Java ME. Ele foi baseado no Swing do Java SE e possui compatibilidade com MIDP 2.0, CDC, PBB e Java SE.

Com o LWUIT, diminui-se muito a necessidade de se desenhar telas em canvas para se obter interfaces amigáveis. É importante saber que o LWUIT não possui compatibilidade com a classe Canvas, portanto não é possível se desenhar interfaces canvas com o LWUIT. Mas oferece componentes visuais muito ricos, que podem ser muito mais atraentes do que interfaces desenhadas em LCDUI ou canvas, com muito menos esforço.

A biblioteca oferece melhorias a componentes já existentes no Java ME, como List, Form, Alert, entre outros. O LWUIT oferece ainda:

- Suporte a touch Screen;
- Diversas fontes;
- Animações;
- Botões;
- Transições de telas animadas;
- Temas, que podem ser incluídos pelos próprios usuários;
- Layouts;
- Utilização de abas, como no Java SE;
- Integração 3D;
- Caixas de diálogo, entre outros.

No LWUIT, os temas podem ser carregados em tempo de execução. Para isto basta carregar um recurso com os temas e escolher um tema para ser o tema corrente da aplicação. Para criar um tema existe uma ferramenta muito interessante, disponibilizada pelo LWUIT toolkit, chamada ResourceEditor.

Para oferecer portabilidade o LWUIT implementa sua própria fina camada no topo do sistema nativo canvas e providencia uma abstração quanto aos diferentes dispositivos. Esta abstração é obtida através de classes chaves que escondem classes específicas, como Graphics, Image e Font.

Na figura 13 vemos uma demonstração da tela do LWUIT.



Figura 13: Tela da aplicação demo do LWUIT.

2.4 JSP (*JavaServer Pages*)

JSP (JavaServer Pages) é uma tecnologia, baseada na plataforma Java, que permite o desenvolvimento de paginas interativas, com acesso a banco de dados, similar a outras tecnologias como ASP (Active Server Pages) ou PHP.

Uma página criada com a tecnologia JSP, depois de instalada em um servidor de aplicação compatível com a tecnologia JEE, é transformado em um servlet, são exemplos de servidores de aplicação compatíveis com Java EE o Tomcat e o Glassfish.

Cada servlet é um objeto Java que recebe requisições (request) e retorna algo (response), como por exemplo, uma página HTML ou uma imagem. Sua principal vantagem é o seu desempenho, pois servlets ficam carregados em memória e a cada cliente que se conecta ao servidor, o servlet cria uma nova thread.

2.5 API Google Maps

A API do Google Maps permite usar Java Script para incorporar o Google Maps em uma pagina web. A API fornece diversos utilitários para manipular e adicionar conteúdo ao mapa por meio de diversos serviços, o que permite criar aplicativos de mapas robustos em uma aplicação web.

A característica que sem dúvida mais chama a atenção do funcionamento de Google Maps é a sua interatividade. Podemos clicar várias vezes no mapa para nos movermos pelo mundo, dar zoom e escolher o tipo do mapa, sem a necessidade de recarregar a página, tudo isso de forma fácil e intuitiva. Este tipo de aplicação web se enquadra dentro da tecnologia que Adaptive Path denominou AJAX (Asynchronous Javascript + XML).

Em um navegador, e com um simples clique o usuário pode dar a volta virtual pelo mundo, desfrutando de fotos aéreas de grande qualidade em algumas zonas, e do mapeamento vetorial completo de outras.

Veja na figura 14 o funcionamento da API Google Maps. Na figura vemos PHP Script, mas este pode ser substituído facilmente por outras linguagens como JSP ou ASP.

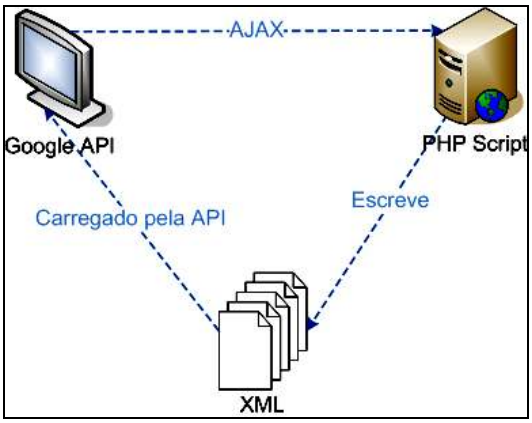

Figura 14: Funcionamento da API Google Maps.

2.6 Api Open Social

O serviço OpenSocial define uma API comum para desenvolver aplicativos sociais que irão funcionar em diversos sites. Elas possibilitam que os desenvolvedores criem aplicativos utilizando *JavaScript* e HTML padrão para acessar amigos de uma rede social e atualizar feeds.

O uso de uma API comum significa que você precisa aprender menos para desenvolver

para diversos sites. O OpenSocial está sendo desenvolvido por um conjunto amplo de membros da comunidade da web. O objetivo final é que qualquer site de relacionamentos seja capaz de implementar a API e hospedar aplicativos sociais de terceiros. Há muitos sites que suportam o OpenSocial, entre eles o hi5, o *LinkedIn*, o MySpace, o Netlog, o Ning, o orkut e o Yahoo!

4. SUGESTÕES FUTURAS E CONCLUSÃO

Com a construção do protótipo foi testada a viabilidade do projeto, o protótipo possibilitou a criação de uma solução para integrar tecnologias móveis e de posicionamento a redes sociais, com ele é possível aumentar a interação dos usuários entre si e sua interação com as novas tecnologias.

No desenvolvimento da arquitetura do projeto foi constatado que é possível implementar o projeto em diversas linguagens e ainda sim elas conversarem entre si.

Uma proposta para trabalhos futuros é que os usuários possam deixar dicas sobre os locais, com algumas informações úteis sobre localidades próximas, e quando o usuário estiver em determinada localidade receber esses dados.

Outra proposta seria aumentar a interação entre usuários, permitindo pelo sistema enviar e receber mensagens de seus contatos próximos. Poderia ser criada também uma rede de relacionamentos nova possibilitando ao usuário conhecer outras pessoas, que não são seus contatos, mas usuários do sistema, receber alguns de seus dados, juntamente com uma foto, como uma opção para aumentar a sua lista de contatos.

5. REFERÊNCIAS

Bonfim Júnior, Francisco Tarcizo JSP : a tecnologia Java na Internet São Paulo: Érica, 2003.

Caelum. Disponível em < www.caelum.com.br >. Acesso em: 28 de Setembro 2010.

Dev Media. Dev Media Group. Disponível em <http://www.devmedia.com.br/>. Acesso em: 21 Agosto 2010.

Disponível em <Rafael Dohms. http://www.rafaeldohms.com.br/>. Acesso em: 21 Agosto 2010.

IBM. Think small with J2ME. Disponível em <http://www.ibm.com/developerworks/library/wi-j2me>. Acesso em: 14 Agosto 2010

JavaFree.org - comunidade java formada pela coolaboração dos desenvolvedores da tecnologia Java com publicações de artigos. Disponível em < www.javafree.org >. Acesso em: 14 Agosto 2010.

Lightweight UI Toolkit < lwuit.dev.java.net >. Acesso em: 28 de Setembro 2010.

Muchow, J. W. Core J2ME - Tecnologia e MIDP. Pearson Makron Books, São Paulo, 2004.

Neto, A . ,M. LWUIT:"Swing" para Java. Java Magazine, Edição 58

Nokia. Nokia on the Web. Disponível em < www.nokia.com >. Acesso em: 21 Agosto 2010.