UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO DE PÓS-GRADUAÇÃO
ESPECIALIZAÇÃO EM DESENVOLVIMENTO WEB

RODOLFO JOSÉ MASSARO

SISTEMA DE POSICIONAMENTO GEOGRÁFICO DE ORELHÕES

MONOGRAFIA DE ESPECIALIZAÇÃO

LONDRINA
2013

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Londrina
Diretoria de Pesquisa e Pós-Graduação
Especialização em Desenvolvimento Web

RODOLFO JOSÉ MASSARO

SISTEMA DE POSICIONAMENTO GEOGRÁFICO DE ORELHÕES

Monografia apresentada como requisito parcial para obtenção do grau de Especialista em Desenvolvimento Web, do Programa de Pós-Graduação em Desenvolvimento Web, da Universidade Tecnológica Federal do Paraná, campus Londrina. Área de Concentração: Sistemas de Informação.

Orientador: Profa. Dra. Ligia Flavia Antunes Batista

LONDRINA
2013

TERMO DE APROVAÇÃO

Título da Monografia

SISTEMA DE POSICIONAMENTO GEOGRÁFICO DE ORELHÕES

por

RODOLFO JOSÉ MASSARO

Esta monografia foi apresentada às 10h30 do dia **22** de **fevereiro** de 2013 como requisito parcial para a obtenção do título de ESPECIALISTA EM DESENVOLVIMENTO WEB. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho _____.

(aprovado, aprovado com restrições ou reprovado)

_____
Prof. Elias Canhadas Genvigir
(UTFPR)

_____
Prof. André Luis dos Santos Domingues
(UTFPR)

_____
Profa. Ligia Flavia Antunes Batista
(UTFPR)

Visto da coordenação:

_____
Prof. Thiago Prado de Campos
Coordenador da esp. em Desenvolvimento Web

_____
Prof. Walmir Eno Pottker
Coordenador de Pós-Graduação Lato Senso

À minha família !

RESUMO

MASSARO, Rodolfo J. **Sistema de posicionamento geográfico de orelhões**. 52f. 2010. Monografia (Pós Graduação em Desenvolvimento Web) – Universidade Tecnológicca Federal do Paraná. Londrina, Paraná, 2013.

A Internet revolucionária das comunicações é coisa do passado. Foi-se o tempo em que ela servia apenas para grandes corporações, empresas ou entusiastas da tecnologia digital, disponibilizarem seus conteúdos ou sistemas on-line. Prova disso é a disseminação das redes sociais e dos sistemas colaborativos onde os usuários geram e compartilham conteúdo interessante para toda a comunidade. Essa característica da internet, despertou interesse por estudar as tecnologias disponíveis, para o desenvolvimento de um tipo de sistema on-line denominado *mashup* onde a combinação de um ou mais ferramentas e conteúdos já disponíveis na rede, cria um aplicativo completamente inovador. Nesse sentido foi pensado numa aplicação que pudesse ajudar as comunidades de Londrina e Tamarana a obter e disseminar conteúdo sobre telefones públicos através de um mapa. Após alguma análise foi possível estabelecer algumas linguagens e tecnologias que mesclavam nosso conhecimento prévio e a massiva aplicação neste tipo de sistema. Buscamos então as informações sobre os telefones públicos que foi gentilmente cedida pela concessionária Sercomtel. De posse de todas as informações necessárias construímos o sistema utilizando como linguagens do lado servidor JAVA e tecnologias auxiliares como JSP para a aplicação e SQL para manipulação de dados via banco de dados relacional. Para o lado cliente foram utilizados HTML, JavaScript e API´s do Google maps, bem como JQuery e um pouco de CSS. Após a construção do sistema, sucederam-se os testes com alguns usuários onde, através de um questionário aplicado logo após os testes pudemos comprovar, embora com algumas ressalvas que o sistema pode ajudar a comunidade dentro do que foi a sua proposta inicial.

Palavras-Chave: Mashup. Sistema Web. Geoprocessamento. Posicionamento Geográfico.

ABSTRACT

MASSARO, Rodolfo J. **Geographical positioning system of public telephones.** 52p. 2013. Monography (Post graduation in Web Development) – Universidade Tecnológica Federal do Paraná. Londrina, Paraná, 2013.

The Internet as a revolutionary of the communications is a thing of the past. Gone are the days when it served only to large corporations, companies or enthusiasts of digital technology, to make available their content or online systems. Proof of this is the spread of the social networks and collaborative systems where users generate and share interesting content for the entire community. This feature of the Internet, has raised interest in studying the available technologies for developing a type of online system called mashup where the combination of one or more tools and content already available on the network, creates a completely new application. In this sense it was thought in an application that could help communities of Londrina and Tamarana to obtain and disseminate content on payphones across a map. After some analysis it was possible to establish some languages and technologies that combined our prior knowledge and massive application in this type of system. We seek then the information about the public telephones which was kindly provided by the concessionaire Sercomtel. Armed with all the information necessary we built the system using as server side language, JAVA and assistive technologies such as JSP to the application, and SQL for data manipulation via relational database. For the client side we used HTML, JavaScript and Google Maps API's, as well as a bit of JQuery and CSS. After the construction of the system, followed up the tests with some users where, via a questionnaire administered immediately after the tests we could prove, with some reservations that the system can help the community in what was its initial proposal.

Keywords: Mashup. Web System. Geoprocessing. Geographic Positioning.

SUMÁRIO

1 INTRODUÇÃO

O surgimento de ferramentas como *Google Maps*, *Google Earth* *WikiMapia* permite que pessoas comuns tenham acesso à qualquer parte do planeta por meio de aplicações que misturam imagens de satélite, modelos 3D e GPS.

Estas aplicações podem ser “embarcadas” gratuitamente em outros sistemas web denominados *mashups* que podem facilitar o acesso a informações de interesse do público geral ou de uma determinada comunidade. Isto tem reforçado o caráter social e colaborativo de utilização da internet.

O telefone público popular orelhão, é uma importante ferramenta no processo de expansão da telefonia fixa e de acesso à informação. O Brasil já conta com mais de um milhão de aparelhos, distribuídos por mais de 5,5 mil municípios; os telefones públicos estão presentes inclusive em localidades onde ainda não há disponibilidade de linhas residenciais segundo a ANATEL (ANATEL).

A ANATEL (Agência Nacional de Telecomunicações), órgão regulador das telecomunicações brasileiras disponibiliza um sistema em seu site (http://sistemas.anatel.gov.br/SGMU/ - SGMU - Sistema de Gestão das metas de universalização) que oferece informações aos usuários sobre os telefones públicos de todo o Brasil. As informações vão desde o tipo do aparelho (comum, adaptado para cadeirante, para deficiente auditivo) até o endereço que esse equipamento se encontra, inclusive suas coordenadas geográficas. Contudo, não há nesse sistema a possibilidade de saber qual o orelhão mais próximo de um endereço. Também não há como reportar para a operadora responsável caso o orelhão esteja com problemas. Alem disso o sistema não é acessível para usuários que utilizam navegadores Firefox ou Google Chrome por exemplo.

2 OBJETIVOS

2.1 OBJETIVO GERAL

Desenvolver um sistema web onde o usuário possa através de um mapa, visualizar o(s) orelhão(ões) perto do endereço de sua preferência, consultar os orelhões por tipo (comum ou adaptado para pessoas com deficiências) visualizar a distribuição geográfica dos orelhões de toda a cidade de Londrina e Tamarana, cadastrar-se e reportar problemas para a concessionária responsável (Sercomtel).

Com isto pretende-se ajudar a concessionária Sercomtel a manter sua planta de aparelhos de telefone público uma vez que é criado um canal mais interativo para que a comunidade contribua informando problemas com os aparelhos.

Outro objetivo que pretende-se atingir é oferecer para a comunidade de Londrina e Tamarana um serviço público inédito e de grande utilidade uma vez que será possível consuta os orelhões adaptados para pessoas com deficiências entre outras funções.

3 REFERENCIAL TEÓRICO

3.1 MASHUP

Segundo (MELO JUNIOR, 2007) uma aplicação Mashup reúne conteúdo de fontes diversas utilizando API´s e técnicas padrões de mercado afim de criar uma nova visão para o usuário. São exemplos os seguintes:

- Fix My Street (http://fixmystreet.com.br/): Oferecem espaço para que qualquer pessoa exerça sua cidadania apontando problemas de infraestrutura nos seus bairros como por exemplo ruas esburacadas e falta de sinalização. Integrando o google maps é possível localizar o endereço, fazer a reclamação e publicar foto.(trecho integral de fix my street);
- Mapeia (http://mapeia.com): Traça a rota por rodovia entre duas cidades informadas pelo usuário e destaca no mapa. Mostra informações de pedágios e calcula valores da viagem de ida e volta.
- WikiCrimes (http://www.wikicrimes.org) : Compila e disponibiliza informações sobre crimes reportados pelos usuários. Densidade, tipos de crimes, principais causas para os crimes da região são exemplos de funcionalidades do site.

Segundo (MERRILL, 2006) um novo modelo de aplicativos de integração de dados por toda a internet desde 2006 (ano de publicação do artigo). Ainda segundo (MERRILL, 2006) popularidade desse tipo de aplicação está no fato da ênfase na participação dos usuários e na maneira como eles agregam os dados de fontes terceiras para gerar seu conteúdo. Para entender melhor um Mashup, basta olhar para a o significado do próprio termo Mashup que, no contexto da música pop, é o nome que se dá a uma nova música obtida através da mistura de faixas vocais e instrumentais de duas músicas diferentes (que geramente pertencem a gêneros diferentes). Assim, trata-se de uma composição incomum e inovadora de conteúdos (muitas vezes obtidas a partir de fontes de dados não relacionados).

3.1.1 MASHUP DE MAPAS

Ainda segundo (MERRILL, 2006), vivemos na era da tecnologia da informação onde as pessoas coletam uma enorme quantidade de dados sobre coisas e atividades e associam esses dados a locais. Esses tipos de dados, por sua característica parecem pedir para serem dispostos graficamente. Neste sentido, um dos grandes catalisadores para o advento dos mashups foi o lançamento da API do Google Maps. Isso permitiu que os desenvolvedores da web, desde os mais amadores pudessem gerar todo tipo de conteúdo georreferenciado e mostrar graficamente. Para não ficar de fora, Microsoft, Yahoo e AOL também lançaram em seguida suas API´s para mapeamento.

Esses tipos de mashups são os mais comuns e os exemplos encontrados na introdução desse trabalho podem descrever melhor essa classificação de mashups.

3.1.2 MASHUP DE VÍDEO E FOTO

Segundo (MERRILL, 2006) o surgimento do alojamento de fotografias e sites de redes sociais como o Flickr com API´s que possibilitam o compartilhamento de fotos levou ao surgimento dessa variedade interessante de mashup. Isso se deve ao fato de que esses provedores de conteúdo têm metadados associados com as imagens que hospedam (como a pessoa que tirou a foto, de que a imagem trata, onde e quando foi tirada e muito mais). Os designers de mashup de fotos podem misturar as fotos com outras informações que podem ser associadas aos metadados. Por exemplo, um mashup poderia analisar letras de música ou poemas afim de poder criar um mosaico de fotos relevantes para esse assunto.

3.1.3 ARQUITETURA DOS MASHUPS

Uma aplicação mashup segundo (MERRILL, 2006) é composta por três partes que são logicamente e fisicamente bem distintas: fornecedores de API/conteúdo, o mashup e o navegador do cliente:

- Os fornecedores de API/Conteúdo: Google, Microsoft, Yahoo são alguns desses fornecedores que, através de suas API´s fornecem as peças (dados) do conteúdo que se está almejando para que o designer do mashup pense na melhor maneira de apresentar esses dados. No exemplo de mashup deste trabalho, os fornecedores são o Google e a operadora Sercomtel. Para facilitar o recolhimento dos dados, os fornecedores frequentemente expõem seu conteúdo através de protocolos Web tais como REST, Web Services.
- O mashup: O lugar onde o quebra-cabeças é montado. Curiosamente, apenas porque este é o lugar onde a lógica de mashup está, não é necessariamente onde ele é executado. Os mashups podem ser implementados da mesma forma que às aplicações Web tradicionais usando do lado do servidor tecnologias de geração de conteúdo dinâmico, como servlets Java, CGI, PHP ou ASP. O conteúdo enxertado, por outro lado pode ser gerado diretamente no navegador do cliente através de JavaScript ou applets, por exemplo. Esta lógica do lado do cliente muitas vezes é a combinação de código incorporado diretamente em páginas de mashup da Web com bibliotecas ou applets de APIs (fornecido pelos provedores de conteúdo) referenciados por essas páginas. Desse modo, eles podem ser chamados de *aplicações ricas para internet* (RIAs), o que significa que eles são mais voltados para a experiência interativa do usuario. (Rich Internet Applications é uma característica da "Web 2.0", a próxima geração de serviços disponíveis na World Wide Web). Os benefícios do lado do cliente incluem menos carga de processamento pois os dados podem ser recuperados diretamente do provedor de conteúdo e uma experiência mais intensa. A API do Google Maps utilizada nesse trabalho é um exemplo de cliente de tecnologia. Muitos mashups utilizam uma combinação de processamento do lado servidor e do lado cliente para atingir seu objetivo. Outros são alimentados diretamente por seus usuários fazendo com que essas informações sejam compiladas e utilizadas para o outro usuário que também alimentará essa base. Outros ainda são resultado da realização de consultas complexas em várias fontes de dados compiladas de forma a atingir seu objetivo.

- Navegador da Web do cliente. É o lugar onde tudo é consumado. Um bom designer deve observar todas as diferenças entre os navegadores atuais do mercado para que seja desenvolvido um produto que se comporte da mesma forma em todos os navegadores.

3.2 GEOPROCESSAMENTO

Segundo (CÂMARA, et al.) Geoprocessamento é a disciplina do conhecimento que utiliza técnicas matemáticas e computacionais para o tratamento da informação geográfica e que vem influenciando de maneira crescente as áreas de cartografia, análise de recursos naturais, transportes, comunicações, energia e planejamento urbano e regional. As ferramentas computacionais para Geoprocessamento, chamadas de *Sistemas de Informação Geográfica (SIG)*, permitem realizar análises complexas, ao integrar dados de diversas fontes e ao criar bancos de dados geo-referenciados.

Nesse intuito o sistema aqui proposto tem por objetivo utilizar as informações de geográficas dos orelhões já armazenadas pela Sercomtel para situá-las utilizando o Google Maps, para prestar um serviço interessante para a comunidade.

3.2.1 EVOLUÇÃO

Segundo (CÂMARA, et al.), os primeiros SIG começaram na década de 60 no Canadá para criar um inventário de recursos naturais, numa época onde as dificuldades eram imensas na área da informática: não existiam monitores de alta resolução, computadores caríssimos e sistemas muito difíceis de usar.

Nos anos 70 adviram novos e mais acessíveis recursos de hardware. Nesse contexto que surgiu a expressão GIS (*Geographic Information System*), também nessa época surgiram os primeiros CAD (*Computed Aided Design*), hoje utilizado em larga escala para todos os tipos de projetos de engenharia o que proporcionou grande evolução para o geoprocessamento.

A década de 80 marca o início de um período que dura até os dias atuais. O IBM PC veio de frente na corrida pelo mercado de computadores domésticos dominado pela Apple, o que configurou o modelo desse mercado como conhecemos

hoje. Paralelamente a isto, nos EUA foram criados centros de pesquisa como o NCGIA *National Centre for Geographical Information and Analysis* marcando o Geoprocessamento como disciplina científica independente.

A partir da década de 90, consolidou-se esse modelo de mercado de computadores e multiplicaram-se rapidamente as velocidades e capacidades dos recursos de hardware das máquinas em progressão geométrica. Isto, associado à evolução dos sistemas de bancos de dados relacionais trouxe proporcionou um alargamento do leque de aplicações de SIG.

4 METODOLOGIA

4.1 PROJETO

4.1.1 REQUISITOS FUNCIONAIS

Segundo (BEZERRA, 2007) requisitos funcionais definem as funcionalidades do sistema. Pela proposta que apresentada pode-se levantar os seguintes requisitos funcionais:

- Visualizar a distribuição geográfica dos orelhões da cidade de Londrina e Tamarana através de um mapa dinâmico e interativo.
- Controlar a visualização dos orelhões por tipo: comum, adaptado para cadeirantes e adaptados para deficiente auditivo.
- Visualizar detalhamento do orelhão.
- Buscar os orelhões mais próximos de um endereço.
- Localizar endereço no mapa.
- Cadastrar-se.
- Efetuar login no sistema.
- Reportar danos com equipamentos.

4.1.2 REQUISITOS NÃO FUNCIONAIS

Segundo (BEZERRA, 2007) requisitos não funcionais declaram as características de qualidade que o sistema deve possuir e que estão relacionadas às suas funcionalidades. Entre eles são descritos confiabilidade, desempenho, portabilidade, segurança, usabilidade.

Para o fim a que se destina esse trabalho, pode-se destacar alguns requisitos não funcionais em detrimento de outros. O desempenho e a confiabilidade são fatores dos quais foi facilmente atingidos bons níveis haja vista a característica acadêmica desse projeto e o elevado nível de maturidade das ferramentas utilizadas, principalmente as API´s do Google. Outros fatores como segurança e a portabilidade não foram abordados pois não se pretende, em primeiro momento a utilização desse sistema na internet.

4.1.3 TECNOLOGIAS ENVOLVIDAS

4.1.3.1 API DO GOOGLE MAPS

Segundo (GOOGLE, 2012) a API do Google Maps é uma biblioteca de elementos em Java Script que permite que o desenvolvedor adicione as ao seu site as funcionalidades e o dinamismo dos mapas do Google Maps. Ela fornece diversas funções para manipulação de adição de conteúdo aos mapas por meio de vários serviços.

Apesar de existirem outras versões, a versão atualmente recomendada para utilização é a versão 3. Ela conta com maior performance e compatibilidade com dispositivos móveis.

O Google coloca à disposição dos desenvolvedores web um tutorial que vai desde a introdução aos conceitos de básicos de utilização até uma referência completa de todos os elementos e possibilidades.

4.1.3.2 LINGUAGENS DE PROGRAMAÇÃO E TECNOLOGIAS

A linguagem utilizada para desenvolvimento da aplicação servidora é JAVA através da plataforma de desenvolvimento (JDK – *Java Development Kit*) versão 1.6. Para o desenvolvimento do provimento das páginas dinâmicas foi utilizado a tecnologia JSP (*Java Server Pages*) associado ao JAVA.

Segundo (ORACLE, 2012) JSP é uma forma conveniente de gerar conteúdo dinâmico em páginas que são disponibilizadas por aplicações web. Esta tecnologia está intimamente associada com tencologia Java Servlet permitindo incluir trechos de código JAVA e chamadas para componentes externos de JAVA dentro do código HTML (ou outro código de marcação como XML).

Ainda no lado servidor, a linguagem utilizada para manipulação de dados foi SQL como estabelecimento impositivo pela utilização de gerenciador de banco de dados relacional My Sql.

No lado cliente, a linguagem de marcação das páginas utilizada foi HTML versão 5.0. Para adicionar dinamismo às paginas do cliente foi utilizado linguagem JavaScript. Para definição dos estilos no lado cliente foi utilizado CSS.

Segundo (W3C) HTML é abreviação de *Hypertext Markup Language* – Linguagem de Marcação de Hypertexto. O HTML é uma linguagem para publicação de conteúdo (texto, imagem, vídeo, áudio e etc) na Web.

JavaScript é uma implementação da ECMAScript que segundo (ECMA) foi originalmente projetada para ser a linguagem de Script da web fornecendo um mecanismo para prover animação em navegadores entre outros.

Segundo trecho traduzido de (W3C) as chamadas folhas de estilo descrevem como documentos são apresentados em monitores, impressoras e talvez, como eles são pronunciados. O consórcio W3C tem promovido ativamente o uso de folhas de estilo na internet desde sua fundação em 1994. Estas atividades da W3C tem produzido algumas recomendações das quais o CSS é o mais amplamente implementado nos browsers. Segundo (W3C) CSS – *Cascade Style Sheet* é um mecanismo simples para adicionar estilos (e.g, fontes, cores, espaços) a documentos web.

Para construção de alguns efeitos foi utilizada uma biblioteca de funcionalidades em JavaScript que tem ganhado cada vez mais destaque na web para tornar o ambiente web nos browsers ainda mais poderoso: o JQuery.

Segundo trecho traduzido de (JQUERY), JQuery é uma pequena biblioteca JavaScript muito rica em recursos. Ela possibilita a manipulação de eventos animação e Ajax de maneira muito mais simples e poderosa e funciona através dos maiores browsers do mercado. JQuery mudou a maneira de muitos programadores, escreverem JavaScript.

4.1.3.3 FERRAMENTAS

A plataforma escolhida para desenvolvimento da aplicação foi o NetBeans 6.9.1 por ser uma plataforma de grande utilização em desenvolvimento de tecnologias JAVA e também por oferecer suporte para o desenvolvimento de todas as outras linguagens de lado cliente necessárias para aplicações web.

Para que a aplicação seja executada e controlada no ambiente web foi utilizada a aplicação GlassFish versão 3, escolhida por ser implementação e recomendação da Oracle (empresa detentora das tecnologias JAVA).

Segundo trecho traduzido disponível em (GLASSFISH) Glassfish é uma aplicação servidora open-source que implementa JAVA EE versão 6 (*Enterprise Edition*). Esta, por sua vez é a plataforma de tecnologias e serviços JAVA tais como JAX-RS 1.1, *JavaServer Faces*(JSF) 2.0, *Enterprise JavaBeans* (EJB) 3.1, *Java Persistence* (JPA) 2.0, utilizadas para o desenvolvimento das aplicações web.

Para implementação do banco de dados foi utilizado o sistema gerenciador de banco de dados MySql versão 5.5. A ferramenta utilizada para modelagem dos dados foi MySql Workbench CE versão 5.2

4.1.3.4 DIAGRAMA DE IMPLANTAÇÃO

Através do diagrama de implantação (figura 1) a seguir pode-se ter uma ideia geral dos artefatos utilizados e suas dependências que compõem a implantação do sistema. O artefato tup_report_bd tem relação de dependência com o artefato MySql SGBD (Sistema geranciador de banco de dados). Da mesma forma o artefato tup_report.war representando o executável do sistema web está para o Container web Glassfish, assim como é dependente do artefato JDK1.6 que representa a máquina virtual Java. Todos esses elementos funcionam em um mesmo servidor representado pelo nó com estereótipo *device.* O cliente conecta-se ao servidor via protocolo TCP/IP conforme característica inerente à internet.

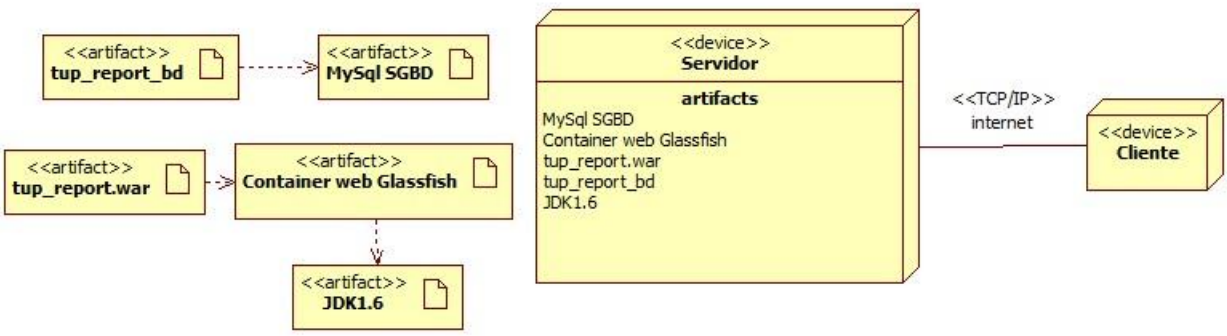


Figura 1 – Diagrama de implantação do sistema

4.1.3.5 DIAGRAMA DE ENTIDADE E RELACIONAMENTO

O diagrama de entidade-relacionamento (figura 2) do projeto é composto de três tabelas: tup_informação que armazena dados da informação que o usuário reportar para o orelhão, tup_usuário para armazenar os acessos dos usuários do sistema e tup_localização para armazenar os orelhões e seus dados como latitude e longitude e suas características.

As tabelas tup_usuario e tup_localização possuem relacionamento de 1 para N com tup_informação. Isso demonstra que cada registro da tabela tup_informacao deverá conter um usuário e um orelhão relacionados, uma vez que uma informação é de um usuário e para um orelhão.

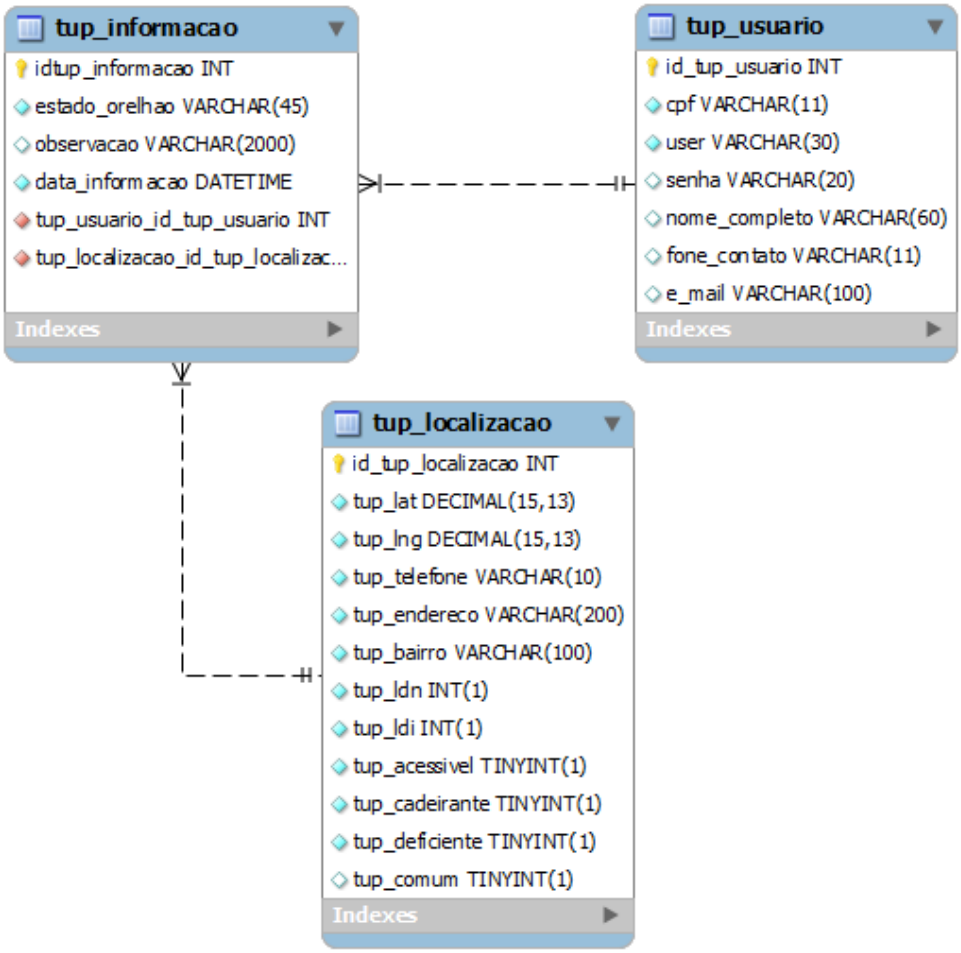


Figura 2 – Diagrama de entidade e relacionamento

4.2 IMPLEMENTAÇÃO

4.2.1 UTILIZAÇÃO DA API DO GOOGLE MAPS

Nesta seção é destacada a utilização da API restringindo-se o foco nos elementos e na forma como estes elementos são aplicáveis ao trabalho. Há outros vários elementos e funcionalidades não citados aqui que fazem parte do conjuto desta API.

Para utilizar a API do Google maps é necessário, pelo menos uma noção básica de JavaScript e uma boa leitura do tutorial formulado pelo google segundo

O principal elemento de qualquer aplicação que utilize a API do Google maps é o próprio mapa e suas operações. Este é o palco para quase toda interatividade do sistema.

Para um melhor entendimento do funcionamento da API, num primeiro momento pode-se dar enfoque para a operação de carga do mapa. Através do diagrama de sequencia da figura 3 é possível obter uma visão geral do processo de carga do mapa e seus elementos envolvidos.

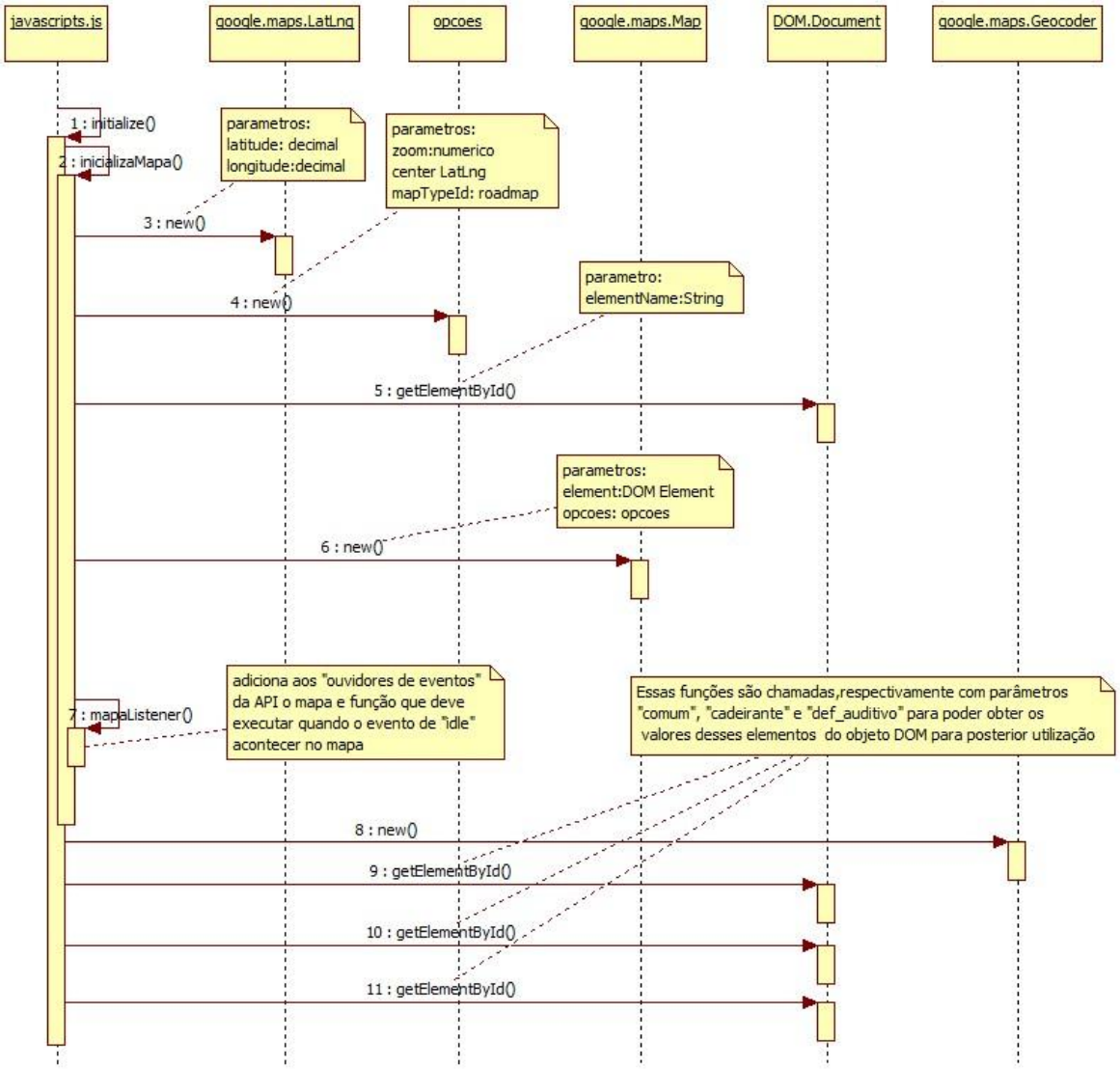


Figura 3 – Diagrama de Sequência carga do mapa

Num segundo momento podemos destacar a sequencia de operações disparadas quando o mapa atinge o estado de “idle” e seu nível de aproximação é tal que justifique uma busca por orelhões conforme figura 4.

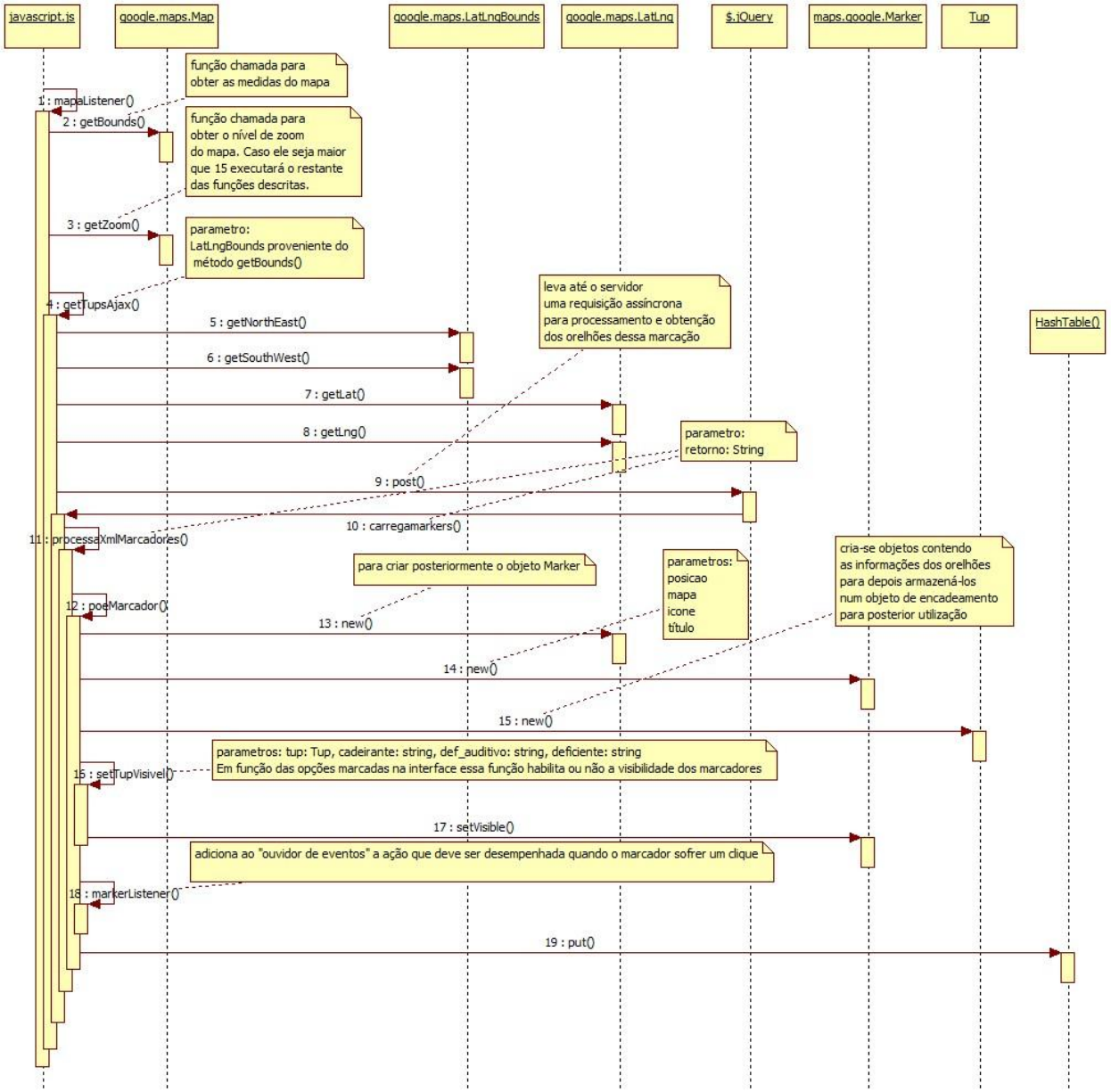


Figura 4 – Diagrama de Sequência carga dos orelhões

A extração do código demonstrado na figura 5 demonstra no sistema a função que inicializa e parametriza o mapa do sistema.

A função chamada inicializaMapa()abriga todas as operações necessárias para que o mapa exista e seja mostrado na tela. A variável latitudeLongitude abriga um objeto google.maps.LatLng reponsável, mais tarde por fazer com que o mapa centralize em conforme com o ponto geográfico passado como parâmetro para essa operação. A variável opcoes cria um objeto que servirá para parametrizar outras

características do mapa como o seu tipo expresso por mapTypeId: google.maps.MapTypeId.ROADMAP o nível de proximidade e a centralização.

```
function inicializaMapa(){
    var latitudeLongitude = new google.maps.LatLng(-
23.292786,-51.173286);
    var opcoes = {
        zoom: 14,
        center: latitudeLongitude,
        mapTypeId: google.maps.MapTypeId.ROADMAP
    }
    mapa = new
google.maps.Map(document.getElementById("map_canvas"),
opcoes);
    mapaListener();

}
```

Figura 5 – Função inicializaMapa

Depois de criar os objetos que contém os parâmetros para inicialização do mapa, parte-se para inicializar o objeto que é o próprio mapa new google.maps.Map(document.getElementById("map_canvas"), opcoes).

Após a criação do mapa utilizou-se outra função mapaListener() (figura 6)para adicionar ao mapa um “ouvidor de eventos” google.maps.event.addListener que, no evento de *idle* (que significa pronto) “dirá” ao mapa que comandos ele precisa executar para que o sistema comece a rodar.

```
function mapaListener(){
    google.maps.event.addListener(mapa, 'idle', function(){
        var lttLngLimites = mapa.getBounds();
        if (mapa.getZoom() > 15){
            getTupsAjax(lttLngLimites);
        }
    });
}
```

Figura 6 – Função mapaListener

No caso conforme a figura 6 a variável lttLngLimites abriga os limites de latitude e longitude em que o mapa está sendo mostrado na tela do cliente através da função mapa.getBounds(). Assim, conforme segue, quando o mapa tiver o zoom maior que 15 (em uma escala estabelecida pelo própria API) mapa.getZoom() > 15

o sistema executará a operação getTupsAjax(lttLngLimites) passando como parâmetro o valor obtido por lttLngLimites .

De acordo com a figura 7 que descreve a função getTupsAjax tem-se as primerias 4 variáveis que carregam os valores de latitude e longitude dos pontos mais a nordeste e sudoeste respectivamente do mapa, provenientes do parâmetro recebido lttLngLimites. São elas latNortheast, lngNortheast, latSouthwest e lngSouthwest. Com esses pontos nordeste e sudoeste definidos, fica fácil saber quais orelhões deve-se trazer do banco de dados da aplicação para mostrá-los no mapa uma vez que cada orelhão possui sua latitude e longitude.

Antes de se buscar pelos orelhões no banco de dados da aplicação, deve-se saber ainda quais orelhões já estão sendo representados no mapa. Dessa forma a variável tups armazena os números de série dos orelhões que já estão sendo mostrados no mapa.

De posse dessas informações, a função, utilizando a tecnologia Jquery (a ser citada mais adiante), envia-as para o servidor atreves da função $.post para o endereço getTups.do para que ele possa tratar essas informações, buscar no banco de dados pelos orelhões que faltam, empacotar essas informações e enviar de volta para o browser do cliente.

```
function getTupsAjax(lttLngLimites){
    var latNortheast = lttLngLimites.getNorthEast().lat();
    var lngNortheast = lttLngLimites.getNorthEast().lng();
    var latSouthwest = lttLngLimites.getSouthWest().lat();
    var lngSouthwest = lttLngLimites.getSouthWest().lng();
    var   tups   =   markersAtivos.keys().length   >   0   ?
markersAtivos.keys().valueOf().toString() : 0;

    $.post('getTups.do', {

        latNortheast: latNortheast,
        lngNortheast: lngNortheast,
        latSouthwest: latSouthwest,
        lngSouthwest: lngSouthwest,
        tups: tups

    }, function carregamarkers(retorno){
        processaXMLMarcadores(retorno);

    });

}
```

Figura 7 – Função getTupsAjax

Após receber a resposta do servidor (figura 7), conforme prossegue a função, é chamada então uma função carregamarkers(retorno) que prossegue chamando a função processaXMLMarcadores(retorno) responsável por processar a resposta e por fim enviá-la para o mapa para ser marcada.

```
function processaXMLMarcadores(responseXML){
    if (responseXML == null) {
        return false;
    } else {
        var                          marcadores                          =
responseXML.getElementsByTagName("marcadores")[0];
        if (marcadores.childNodes.length > 0){
            for  (index  =  0;  index  <  marcadores.childNodes.length;
index++) {
                var marcador = marcadores.childNodes[index];
                var                          id                          =
marcador.getElementsByTagName("id")[0].textContent;
                var                          lt                          =
marcador.getElementsByTagName("lt")[0].textContent;
                var                          lg                          =
marcador.getElementsByTagName("lg")[0].textContent;
                var                          cd                          =
marcador.getElementsByTagName("cd")[0].textContent;
                var                          df                          =
marcador.getElementsByTagName("df")[0].textContent;
                var                          cm                          =
marcador.getElementsByTagName("cm")[0].textContent;
                poeMarcador(id, lt, lg, cd, df, cm);
            }
        }
    }
}
```

Figura 8 – Função processaXmlMarcadores

A função processaXMLMarcadores expressa na figura 8 começa por verificar se a resposta do servidor é nula, neste caso ela não executa o processamento necessário. Esta função tem esse nome pois a resposta esperada do servidor é um XML e, assim sendo, ela deverá efetuar uma varredura nesse código transformando toda informação lá constante em marcadores que serão inseridos no mapa.

As variáveis id, lt, lg, cd, df e cm carregam informações provenientes do servidor respectivamente numero de série do orelhão, latitude, longitude, se o orelhão é adaptado para cadeirantes, se é adaptado para deficiente físico e se o orelhão é comum. De posse dessas informações a função chama poeMarcador na figura 9 para que os orelhões apareçam no mapa. Essa operação é repetida até que todas as tags do XML sejam lidas.

```
function poeMarcador(id, lt, lg, cd, df, cm){
    var latFloat = parseFloat(lt);
    var lngFloat = parseFloat(lg);
    var           latitudeLongitude           =           new
google.maps.LatLng(latFloat,lngFloat);
    var image = 'img/orelhao.gif';
    var marker = new google.maps.Marker({
        position: latitudeLongitude,
        map: mapa,
        icon: image,
        title: id
    });
    var tup = new Tup(id,cd,df,cm,marker);
    setTupVisivel(tup, comum, cadeirante, def_auditivo);
    markerListener(id, marker);
    markersAtivos.put(id,tup);
}
```

Figura 9 – Função poeMarcador

Conforme a figura 9, a função poeMarcador prepara os parâmetros para que eles virem objetos marcadores e populem o mapa. As variáveis latFloat e lngFloat carregam, respectivamente, a latitude e longitude do novo marcador que irá representar um orelhão no mapa. A variável latitudeLongitude carrega um novo objeto google.maps.LatLng utilizado para definir pelo padrão da API, a configuração de latitude e longitude que será aplicada, logo mais ao novo orelhão no mapa. A variável marker refere-se ao objeto google.maps.Marker que é criado pela passagem dos parâmetros de posição, mapa, ícone e título. Esse obejto é o próprio marcador que será colocado no mapa

Após criar o marcador que será introduzido ao mapa, é necessário mostrá-lo segundo alguns critérios definidos pelos requisitos do próprio sistema: os filtros de orelhões que definem quais orelhões que devem ser mostrados entre os orelhões comuns, adaptados para cadeirantes e para deficientes auditivos. A função setTupVisivel (figura 10) verifica se as variáveis com, cad, df_aud recebidas como parâmetro tem o valor “true” para que, nesse caso o marcador referente ao orelhão seja mostrado no mapa através da função setVisible.

```
function setTupVisivel (tup, com, cad, df_aud){
    if (tup.cm == "true"){
        tup.mk.setVisible(com);
    }else{
        if(tup.cd == "true"){
            tup.mk.setVisible(cad);
        }
        if(tup.df == "true"){
            tup.mk.setVisible(df_aud);
        }
    }
}
```

Figura 10 – Função setTupVisivel

Seguindo a sequência da função poeMarcador (figura 9) é chamada a função markerListener (figura 11) responsável por adicionar ao marcador outro “ouvidor de eventos” para obedecer a um requisito funcional do sistema que o marcador apresente algumas informações quando for clicado chamando assim a função getTupDetalheAjax que será abordada mais adiante.

```
function markerListener(id, marcador){

google.maps.event.addListener(marcador,'click',function(){
        getTupDetalheAjax(id, marcador);
    });
}
```

Figura 11 – Função markerListener

A função getTupDetalheAjax (figura 12) recebe como parâmetro o número de série do orelhão e o marcador correspondente. Ela é disparada quando o usuário clica num marcador.

```
function getTupDetalheAjax(id, marcador){
    $.post('getTupDetalhe.do', {
        id: id
    }, function (retorno){
        var tupDetalhe = processaXMLTupDetalhe(retorno);
        var html = processaHTMLTupDetalhe(tupDetalhe);
        var infowindow = new google.maps.InfoWindow({
            content: html
        });
        infowindow.open(mapa,marcador);
    });

}
```

Figura 12 – Função getTupDetalheAjax

Assim como todas as requisições de recursos ao servidor, quando esta função é acionada, é feita uma requisição assíncrona para o servidor utilizando método *post* e a tecnologia AJAX. Dessa forma evita-se que a página tenha que carregar novamente, toda vez que solicite recursos ao servidor. O parâmentro getTupDetalhe.do indica o nome da ação que o servidor vai executar. O próximo objeto carrega o parâmetro da ação (no caso o id do orelhão) para que o servidor saiba qual orelhão vai detalhar.

O terceiro parâmetro da requisição assíncrona é uma a função que vai ser executada quando houver o retorno da requisição. Esta função parametrizada com um objeto de retorno da função entrega esse retorno para processaXMLTupDetalhe que o transforma em um objeto com uma série de marcadores e por sua vez é passado para processaHTMLTupDetalhe para codificar o HTML. Na sequencia é invocado um objeto InfoWindow e, em seguida acionada a função para a abertura da janela, recebendo como parâmetro o mapa e o marcador do orelhão.

4.2.2 FUNCIONALIDADES IMPLEMENTADAS

4.2.2.1 DISTRIBUIÇÃO DOS ORELHÕES PELO MAPA

Dentre as funcionalidades implementadas, a principal é a distribuição visual dos orelhões através de marcadores na API do Google Maps. A ferramenta de mapas disposta pela Google funciona de maneira dinâmica pelo clique e arrasto do mapa pelo frame disponível em qualquer direção que desejar. Também é possível

aumentar e diminuir o enquadramento através do botão central do mouse ou então utilizar os controles no canto superior esquerdo do frame conforme figura 13.

A utilização dos controles inclui ainda uma funcionalidade muito interessante do Google que é o *street view* (figura14), onde é possível passear virtualmente pelas ruas e avenidas da cidade tendo a visão do pedestre. O *street view* pode ser obtido com arrasto do bonequinho no canto superior esquerdo da figura 13 sobre qualquer uma das ruas e avenidas da cidade, se houver visualização terrestre disponível, este logradouro aparecerá em azul.

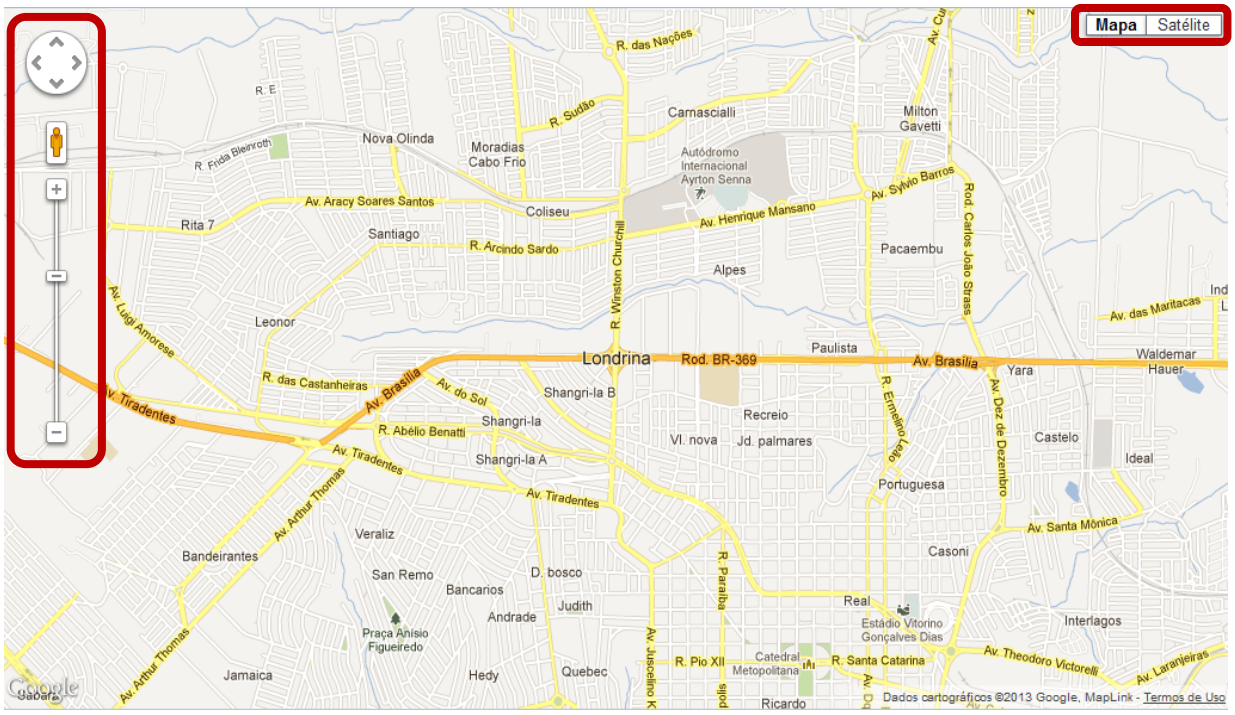


Figura 13 – Figura demonstrando o dinamismo do mapa com os controles

Figura 14 – Figura demonstrando a visão do street view com seus controles

Sobre as funcionalidades do mapa ainda podemos destacar os controles na parte superior direita do *frame* figura 13. Nesta parte há dois botões conforme a figura a seguir onde se mostram dois botões sendo “Mapa” e “Satélite”, os quais, respectivamente alternam a visão do frame entre mapa e a foto satelital do local, ambas as visões em total consonância com a realidade.

Quando está selecionada a visão “Mapa” (figura 15) tem-se uma escala de aproximação e distanciamento do mapa, bem aumentada. A navegabilidade fica bem interessante seja em qualquer nível de detalhamento do mapa que se queira mostrar. Quando está selecionada a opção “Terreno” (figura 16) tem-se uma diminuição da escala de aproximação do mapa e um destaque para o relevo do terreno onde as áreas mais escuras representam as baixadas e as mais claras representam as altitudes.

Quando está selecionada a opção “Satélite” tem-se uma visão da composição de fotos capturadas por satélite da região onde pode-se visualizar as áreas de vegetação e urbanização. Esta visão ainda permite marcar e desmarcar (figura 17 e 18) a opção “Marcadores” que representa a visualização do traçado das ruas e estradas no mapa, bem como seus nomes.

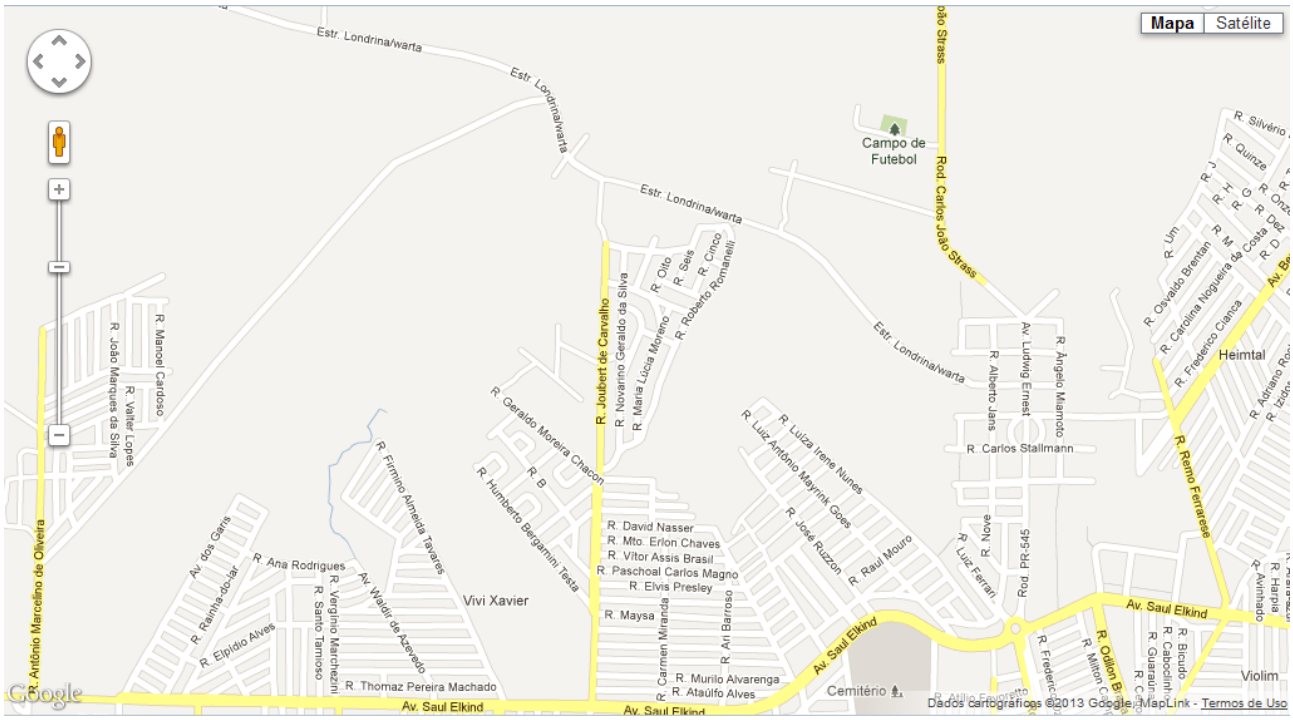


Figura 15 – Figura demonstrando a visão do mapa sem terreno

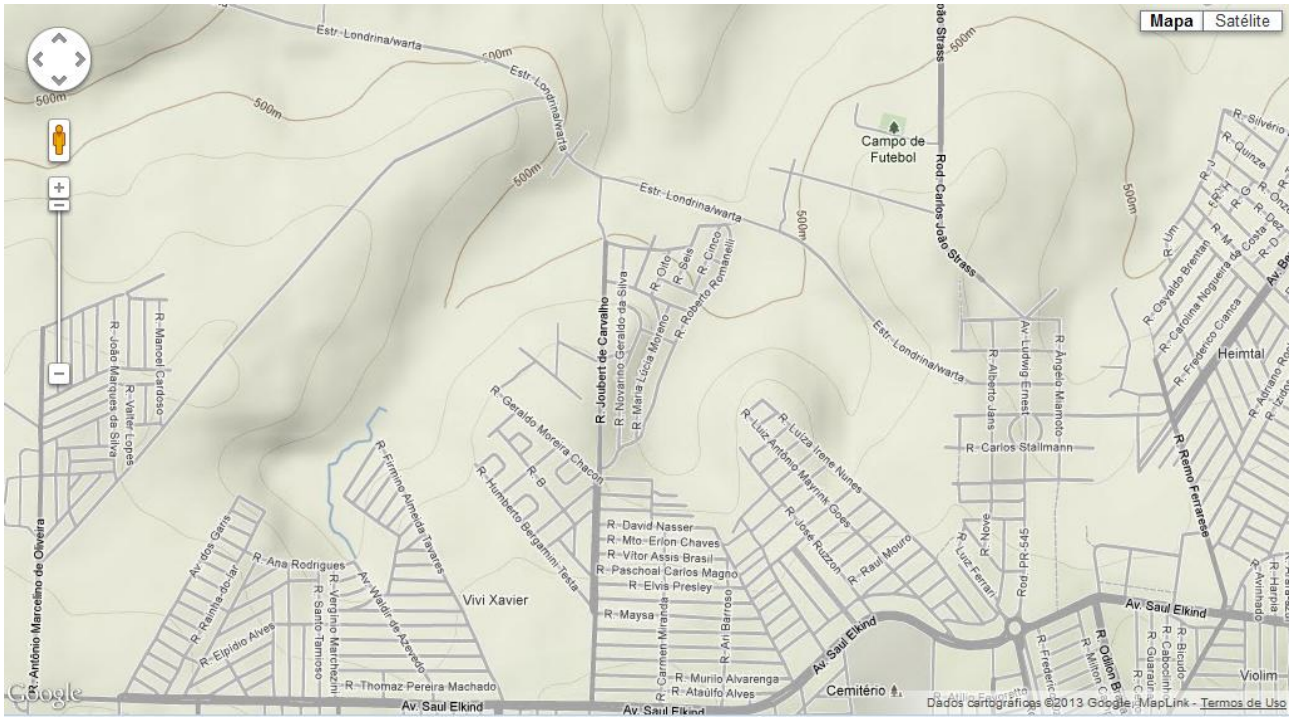


Figura 16 – Figura demonstrando a visão do mapa com terreno

Figura 17 – Figura demonstrando a visão do satélite com marcadores

Figura 18 – Figura demonstrando a visão do satélite sem marcadores

A cada intervenção do usuário no mapa é gerado uma série de eventos que podem ser capturados. Esses eventos capturados fornecem informações importantes sobre a situação do mapa, seu enquadramento, o nível de aproximação e detalhamento, bem como uma série de outras informações.

Assim sendo, pela captura dessas informações é que funciona o mecanismo que dispara a carga dos marcadores representando os orelhões pelo mapa. A cada evento de *"idle"* que representa o estado de "pronto" do mapa, é verificado o nível de zoom e, se ele estiver mais aproximado é disparado o processo de população dos orelhões na tela. Isto faz com que apenas os marcadores daquele determinado enquadramento sejam processados. Desta forma, não se sobrecarrega o sistema com o processamento eventual de todos os marcadores ao mesmo tempo quando o enquadramento do mapa envolver toda a cidade de londrina.

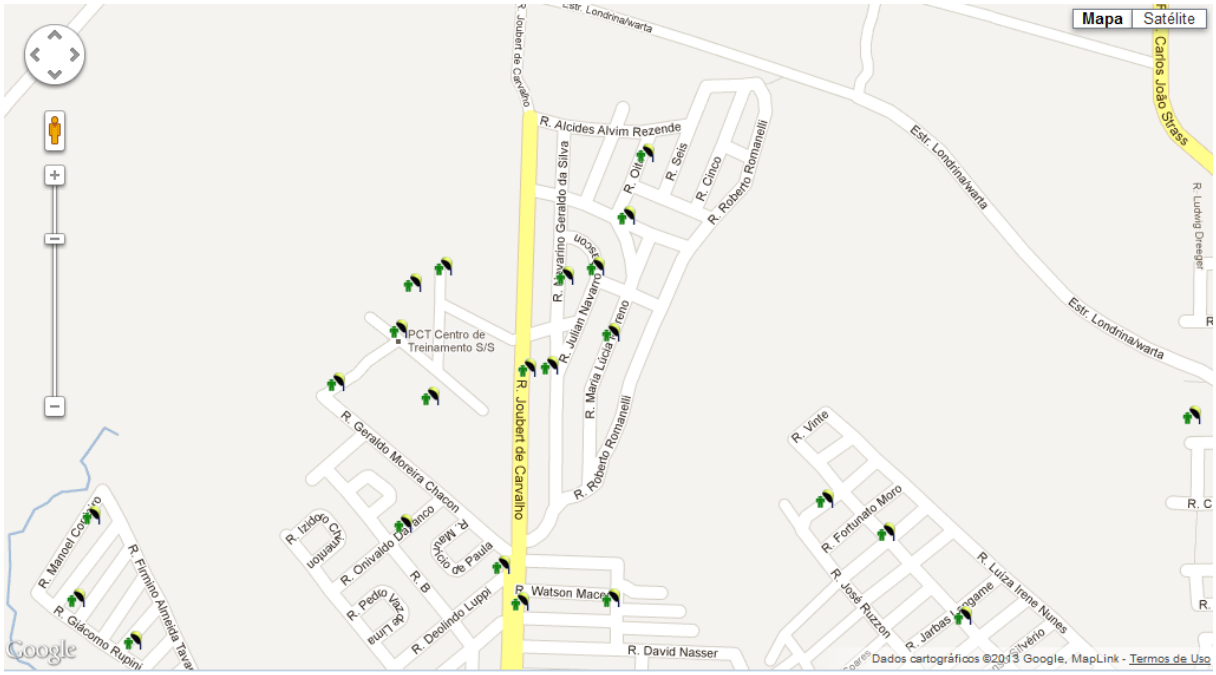


Figura 19 – Figura demonstrando o enquadramento necessário para que sejma mostrados todos os elementos do mapa

4.2.2.2 OBTER INFORMAÇÕES SOBRE DETERMINADO ORELHÃO

Outra funcionalidade interessante do sistema é a obtenção das informações sobre cada orelhão de maneira dinâmica conforme figura 20.

Assim como todas as funcionalidades implementadas que utilizam o banco de dados da aplicação a obtenção de informações sobre o orelhão utiliza tecnologia AJAX para funcionar. O *browser* do cliente faz uma requisição assíncrona para o

servidor e este responde com o detalhamento do orelhão em XML. Uma função transforma esse detalhamento em HTML que por sua vez será o conteúdo da janela de detalhamento do marcador.

Várias janelas podem ser abertas ao mesmo tempo e para cada uma delas todo o processo se repete conforme figura 21.

Figura 20 – Figura demonstrando uma janela de detalhamento do orelhão

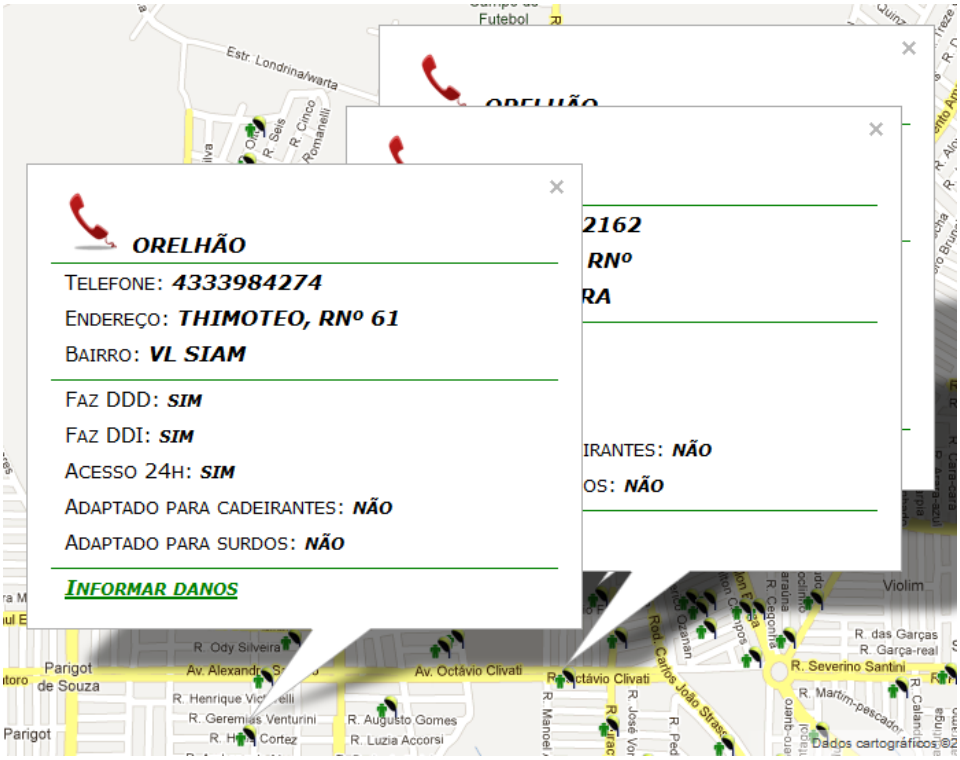


Figura 21 – Figura demonstrando várias janelas de detalhamento de orelhões

4.2.2.3 VISUALIZAR ORELHÕES POR TIPO

Esta funcionalidade (figuras 22, 23 e 24) permite que sejam mostrados os orelhões por tipo: comum, cadeirante, deficiente auditivo. Isto permite que o usuário possa verificar qual o orelhão mais próximo de sua residência que atenda as suas necessidades, assim como poder cobrar as autoridades caso a densidade de aparelhos não seja compatível com as necessidades da população.

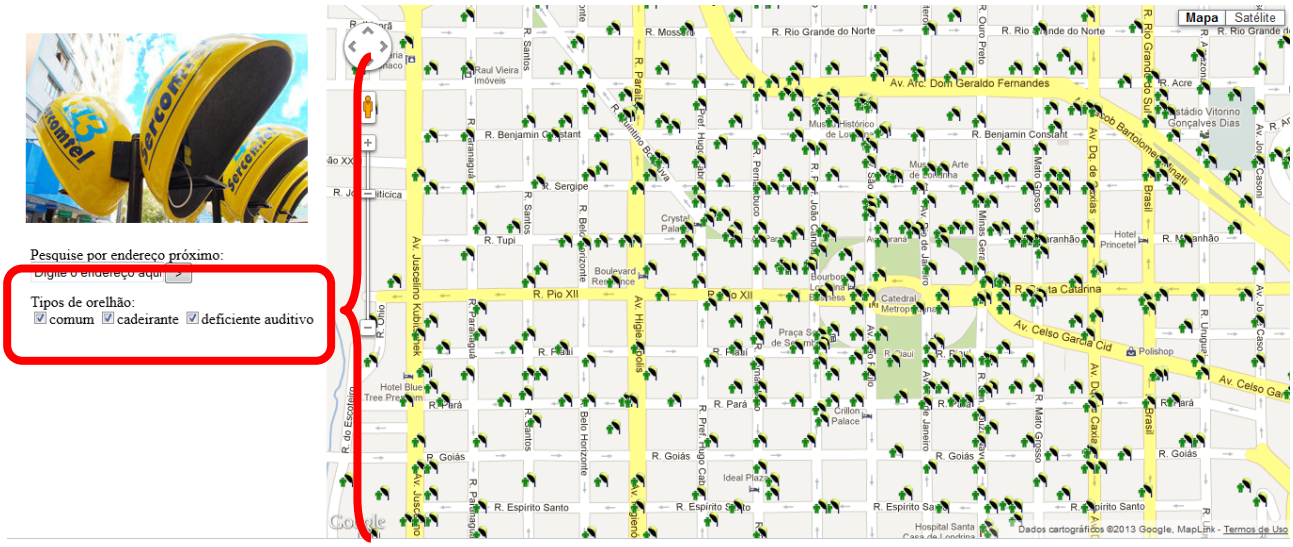


Figura 22 – Figura demonstrando os orelhões distribuídos por tipo (todos marcados)

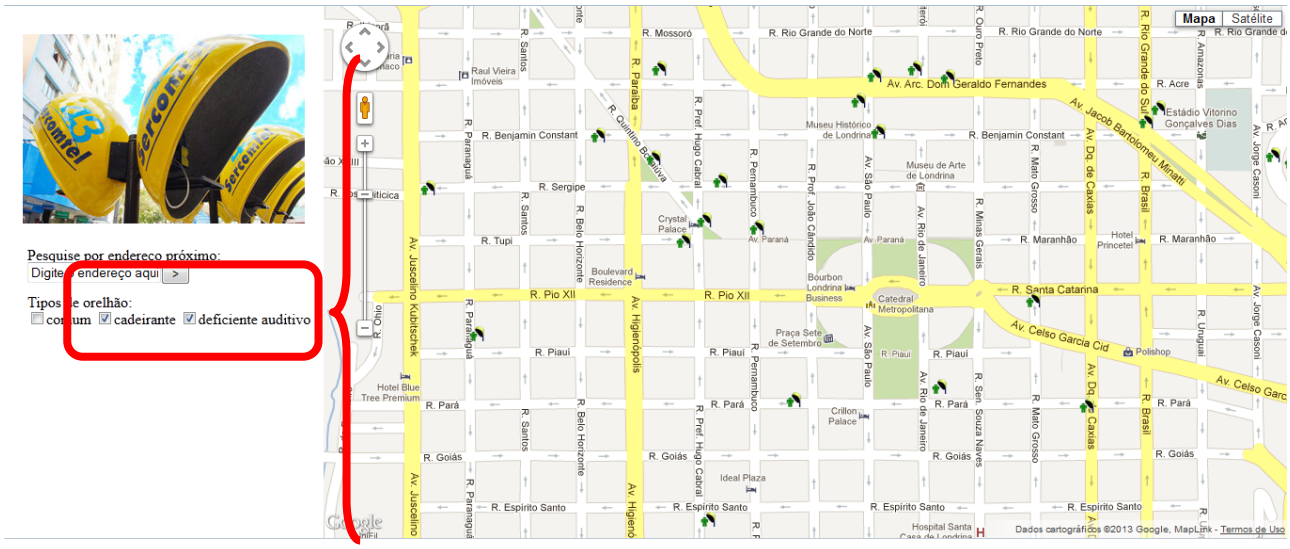


Figura 23 – Figura demonstrando os orelhões distribuídos por tipo (cadeirante e deficiente auditivo marcados)

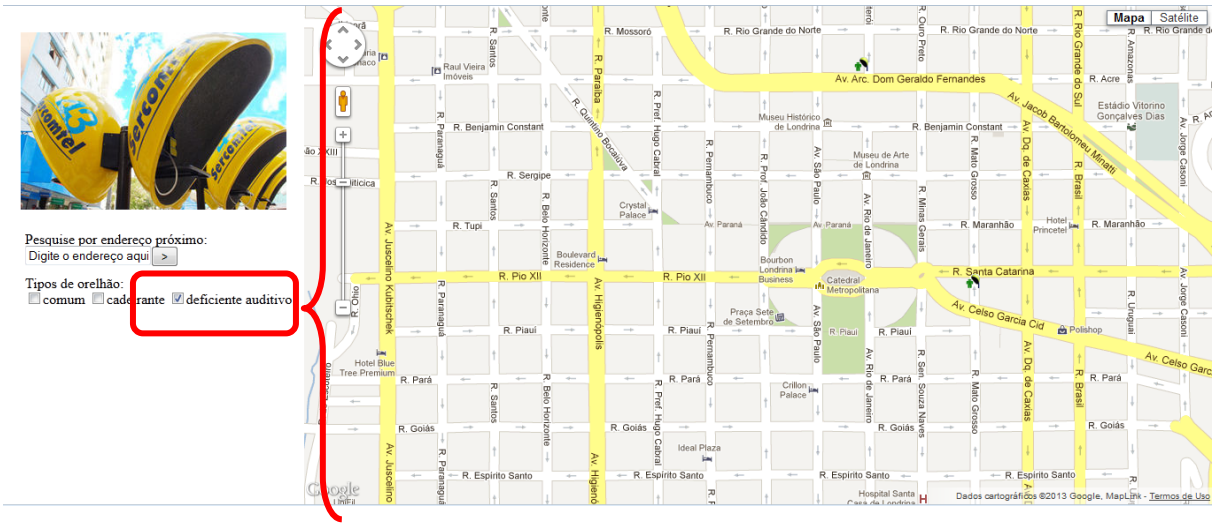


Figura 24 – Figura demonstrando os orelhões distribuídos por tipo (deficiente auditivo marcado)

4.2.2.4 PESQUISAR ENDEREÇO

A funcionalidade de busca de endereços também é oferecida pela API do Google Maps de modo que se digite qualquer parte de um endereço (figura 25) e seja buscado a partir das redondezas de onde está enquadrado o mapa ou adicionando-se aos critérios de busca a localidade determinada onde se quer buscar o endereço.

Isto acontece pela informação do endereço na seguinte caixa de diálogo conforme a figura a seguir:

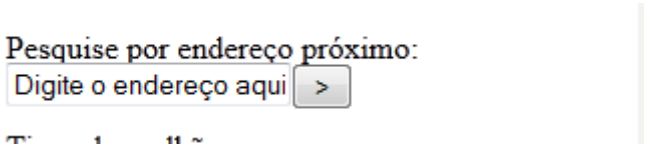


Figura 25 – Figura demonstrando a caixa de digitação de endereços

A utilização da API no caso da busca de endereço pode ser expressa conforme trecho (figura 27) da codificação em destaque a seguir:

```
function codificaEndereco() {
    var endereco =
document.getElementById("pesquisa_endereco").value + ',
Londrina';
    var geoRequest = {
        'address':endereco,
        'region': 'BR'
    };
    geocoder.geocode(geoRequest, function(results, status){
        if (status == google.maps.GeocoderStatus.OK) {
            preencheEnderecos(results);
        } else {
            alert("Não foi possível localizar endereço com os
dados fornecidos: " + status);
        }

    });
    mostraDiv('#caixa_enderecos');
}
```

Figura 27 – Figura demonstrando a função codificaEndereço

Quando o botão de busca é pressionado na interface, é disparada a função codificaEndereco conforme a figura 27. Nesta função a variável endereco recebe o

valor digitado no componente de texto da interface e adiciona a palavra Londrina antecedida por uma vírgula. Isto para limitar mais as buscas na região de londrina, uma vez que um endereço pode se repetir em mais de uma cidade, e isto é bem comum por sinal.

Após a preparação do endereço é criado um objeto geoRequest que conterá parametrizações para a busca do endereço. É importante salientar que conforme se trabalha com esse objeto de “parametrizações” de busca de endereços, pode-se obter pesquisas mais restritas de endereços, ou mais abertas fazendo com que os endereços fornecidos possam ser de lugares indesejados. No caso deste trabalho, foi suficiente a concatenação da palavra Londrina em todas as buscas de endereço e a limitação da região em BR (Brasil) para que os endereços encontrados correspondessem à cidade de Londrina. De outro modo, é possível fornecer por exemplo, a latitude e longitude ou um sensor que demarcaria no mapa o início da pesquisa, bem como a linguagem e outros componentes que poderiam especificar melhor a busca por endereços.

A função que executa a busca propriamente de endereços é a função geocode que é chamada num objeto armazenado pela variável geocoder anteriormente inicializada no código. Esta função recebe como parâmetro o objeto geoRequest comentado acima e outra função que recebe como parâmetro o resultado da pesquisa representado pela variável results e o status que armazena o status resultante da execução da função. Dessa forma, conforme o trecho do código em seguida se o status for “ok” executa-se uma função chamada preencheEnderecos que tem como objetivo montar a codificação HTML dentro de uma divisória com os endereços encontrados aparecendo conforme figura 28.

Figura 28 – Figura demonstrando a caixa de endereços localizados

A API de geolocalização devolve a resposta em formato XML com todas as informações do endereço, inclusive sua latitude e longitude. Portanto, ao clicar em um dos endereços mostrados na divisória, pela captura de sua latitude e longitude, fica fácil localizá-lo no mapa como mostra a figura 29

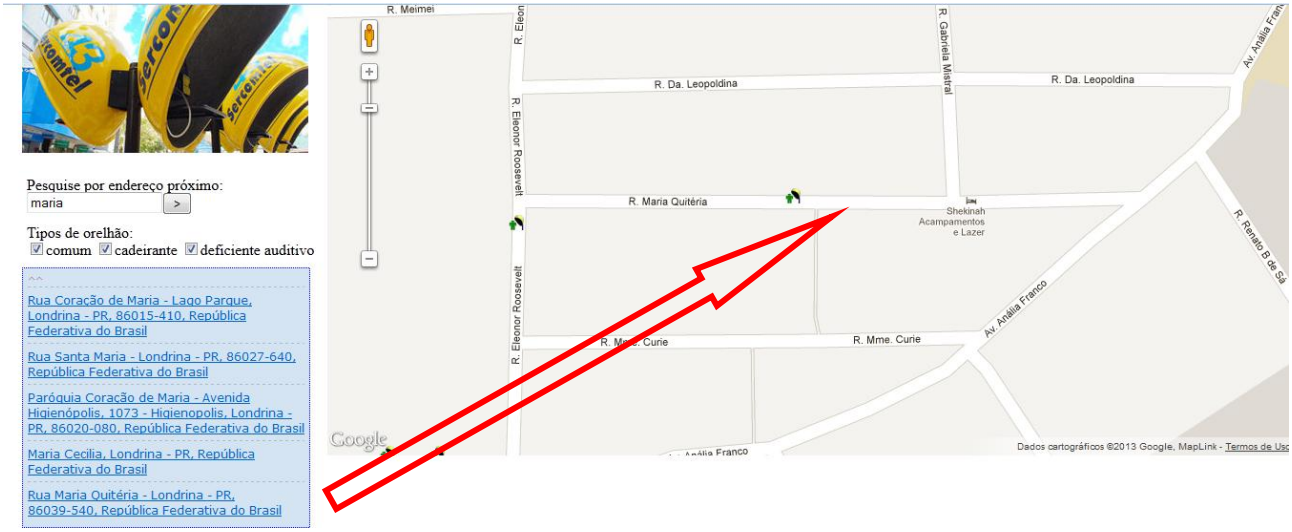


Figura 29 – Figura demonstrando um endereço localizado no mapa

4.2.3 CONSULTAS

4.2.3.1 CONSULTA TODOS OS ORELHÕES DE DETERMINADO QUADRANTE

Esta consulta (figura 31) retorna o os orelhões de determinado quadrante. Ela é utilizada quando o sistema faz uma requisição assíncrona ao servidor para alimentar o mapa com os orelhões do quadrante que ele está focalizando. Este quadrante é marcado pelas latitudes e longitudes dos pontos superior direito e inferior esquerdo do mapa como mostra a figura 30.

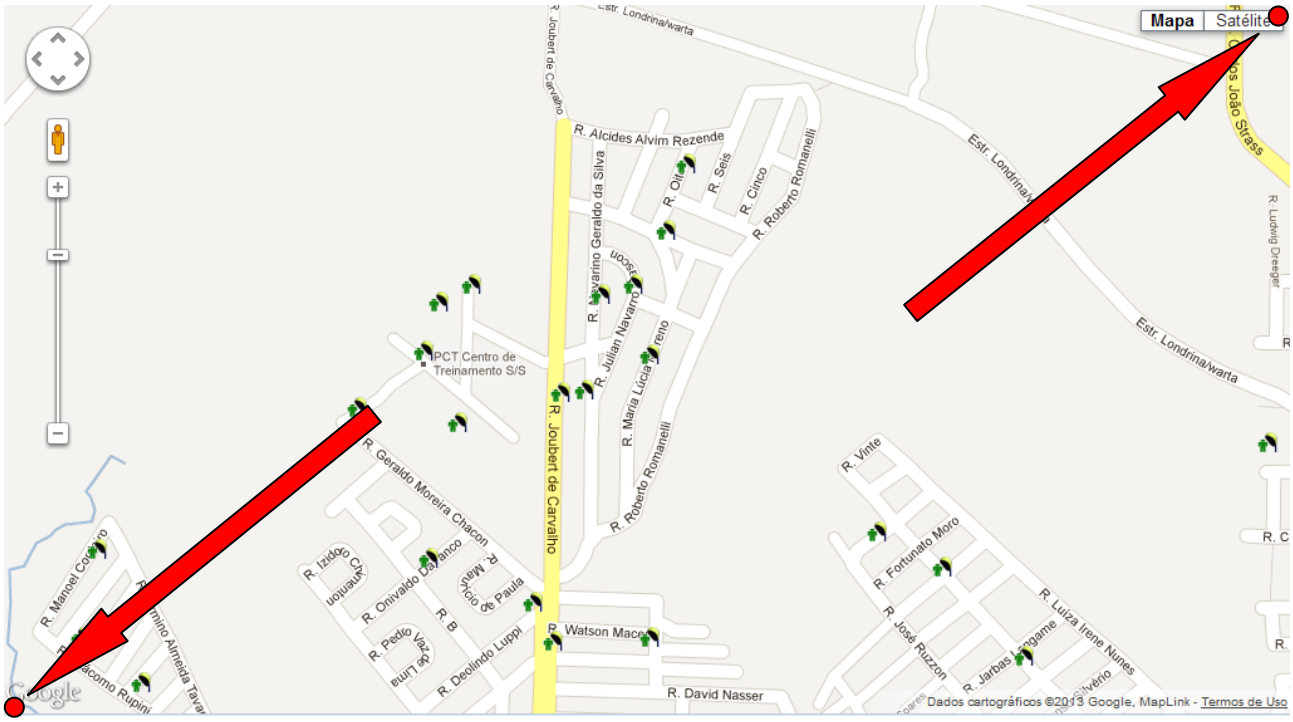


Figura 30 – Figura demonstrando os pontos utilizados para enquadramento do mapa.

```
select
      t.id_tup_localizacao,
      t.tup_lat,
      t.tup_lng,
      tup_cadeirante,
      tup_deficiente,
      tup_comum
from
      tup_report.tup_localizacao t
where
      t.tup_lat between <latitude sudoeste>
      and <latitude nordeste>
      and t.tup_lng between <longitude sudoeste>
      and <longitude nordeste>
      and id_tup_localizacao not in (<orelhões já mostrados>)
```

Figura 31 – Figura demonstrando a instrução de select para capturar os equipamentos entre os pontos do mapa

Quando alguma área do mapa já foi preenchida com orelhões, estes são passados como parâmetro na requisição. Esses orelhões são portanto utilizados na consulta como argumento para que não sejam selecionados novamente, uma vez que já estão marcados no mapa.

4.2.3.2 CONSULTA DADOS DE UM ORELHÃO

Esta é uma consulta simples de detalhamento de um orelhão utilizada quando o usuário clica em um orelhão no mapa. Nesse momento o sistema faz uma requisição assíncrona passando como parâmetro o identificador do orelhão clicado conforme figura 32.

```
select
     t.id_tup_localizacao,        t.tup_lat,        t.tup_lng,
     t.tup_telefone, t.tup_endereco, t.tup_bairro,
     t.tup_ldn, t.tup_ldi, t.tup_acessivel, t.tup_cadeirante,
     t.tup_deficiente
from
     tup_report.tup_localizacao t
where
     t.id_tup_localizacao = <id orelhão>
```

Figura 32 – Figura demonstrando a instrução de select para capturar os dados de um aparelho

4.2.3.3 CONSULTA LOGIN

Consulta destinada a logar o usuário no sistema. Esta funcionalidade foi implementada mas não está em funcionamento atualmente.

```
select id_tup_usuario, cpf, nome_completo, fone_contato,
e_mail from tup_usuario where user = <user> and senha =
<senha>
```

Figura 33 – Figura demonstrando a instrução de select para logar no sistema

4.3 OBTENÇÃO DOS DADOS SOBRE OS ORELHÕES

Atualmente a ANATEL dispõe em seu site dados sobre todos os orelhões de londrina organizados por área de concessão diretamente em http://sistemas.anatel.gov.br/sgmu/Localidade/Lista/frmConsulta.asp?SISQSmodulo=17509 conforme as figuras abaixo.

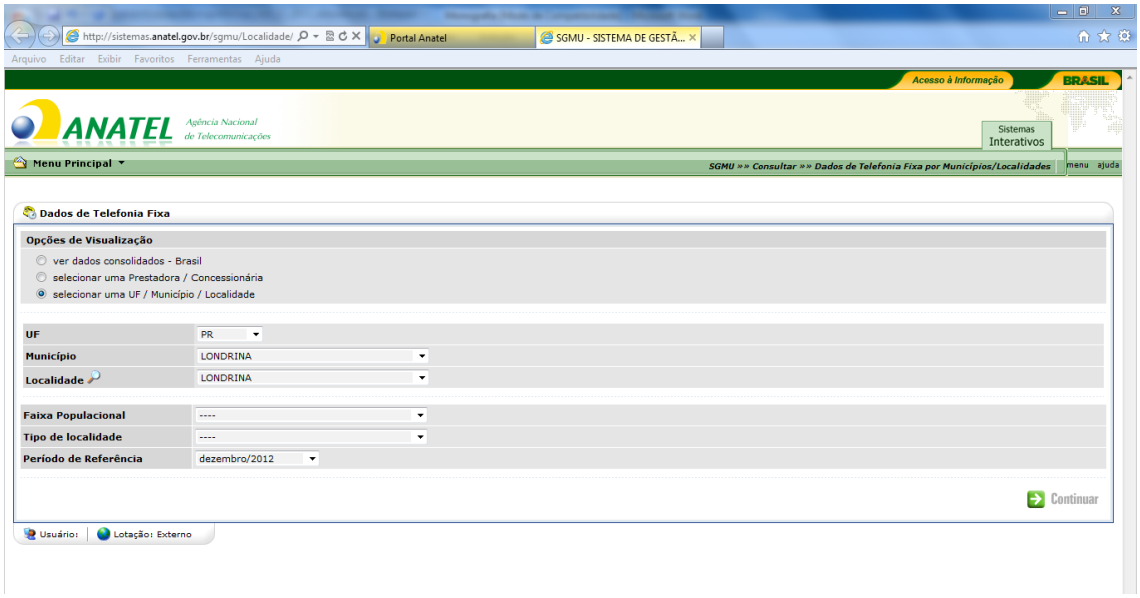


Figura 34 – Figura demonstrando os passos para obtenção das informações sobre os orelhões no site da ANATEL

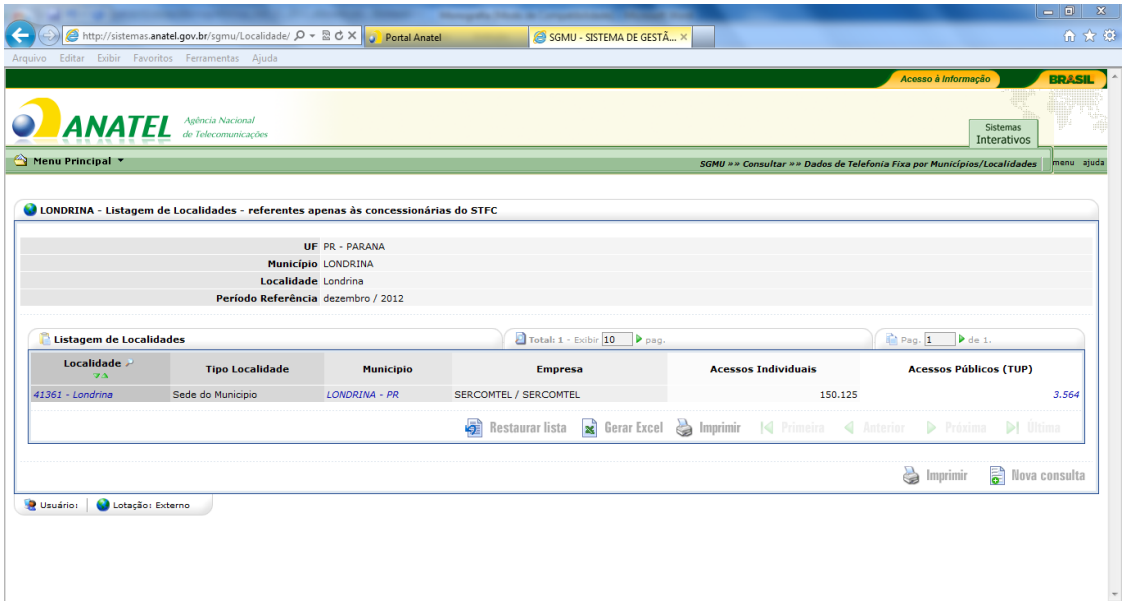
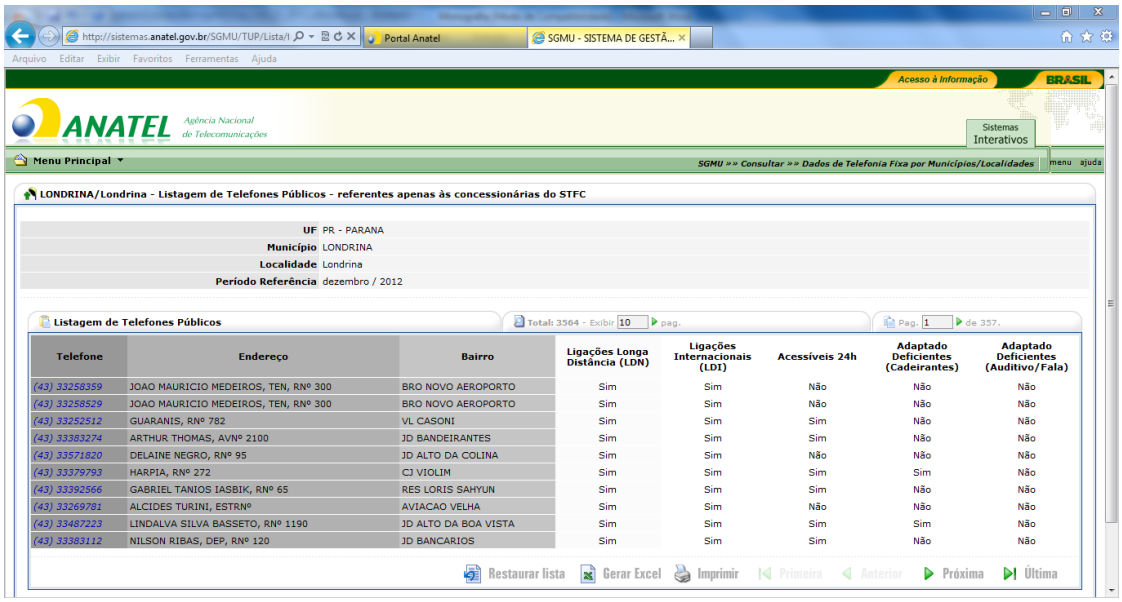


Figura 35 – Figura demonstrando os passos para obtenção das informações sobre os orelhões no site da ANATEL



Telefone	Endereço	Bairro	Ligações Longa Distância (LDN)	Ligações Internacionais (LDI)	Acessíveis 24h	Adaptado Deficientes (Cadeirantes)	Adaptado Deficientes (Auditivo/Fala)
(43) 33258359	JOAO MAURICIO MEDEIROS, TEN, RNº 300	BRO NOVO AEROPORTO	Sim	Sim	Não	Não	Não
(43) 33258529	JOAO MAURICIO MEDEIROS, TEN, RNº 300	BRO NOVO AEROPORTO	Sim	Sim	Não	Não	Não
(43) 33252512	GUARANIS, RNº 782	VL CASONI	Sim	Sim	Sim	Não	Não
(43) 33383274	ARTHUR THOMAS, AVNº 2100	JD BANDEIRANTES	Sim	Sim	Sim	Não	Não
(43) 33571820	DELAINE NEGRO, RNº 95	JD ALTO DA COLINA	Sim	Sim	Não	Não	Não
(43) 33379793	HARPIA, RNº 272	CJ VIOLIM	Sim	Sim	Sim	Sim	Não
(43) 33392566	GABRIEL TANIOS IASBIK, RNº 65	RES LORIS SAHYUN	Sim	Sim	Sim	Não	Não
(43) 33269781	ALCIDES TURINI, ESTRNº	AVIACAO VELHA	Sim	Sim	Não	Não	Não
(43) 33487223	LINDALVA SILVA BASSETO, RNº 1190	JD ALTO DA BOA VISTA	Sim	Sim	Sim	Sim	Não
(43) 33383112	NILSON RIBAS, DEP, RNº 120	JD BANCARIOS	Sim	Sim	Sim	Não	Não

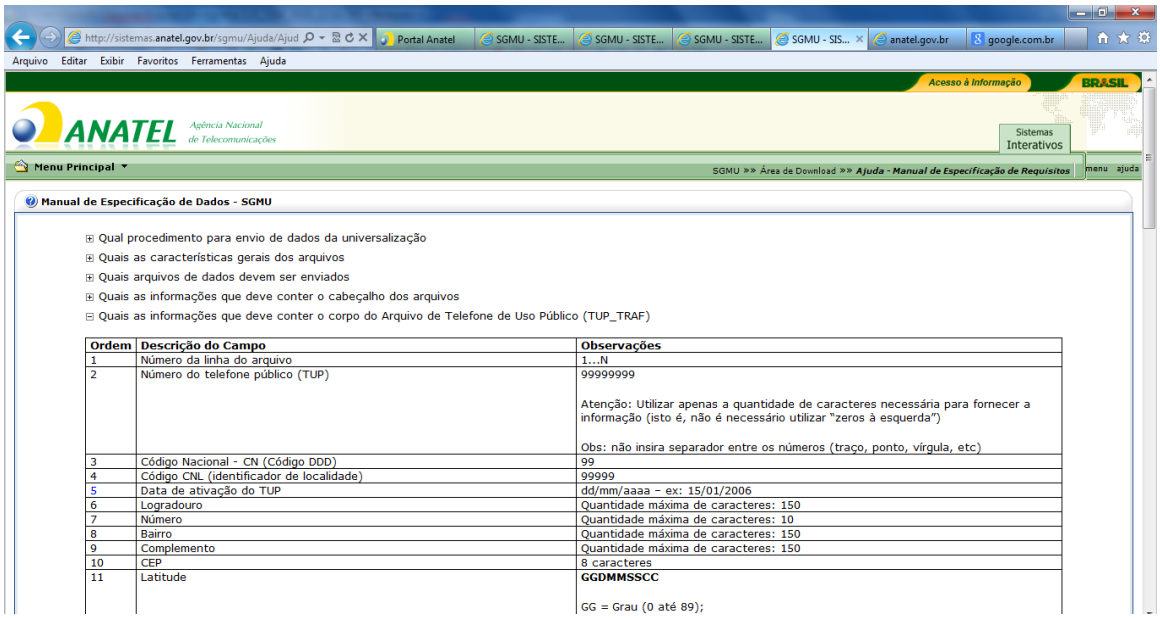
Figura 36 – Figura demonstrando os passos para obtenção das informações sobre os orelhões no site da ANATEL

Conforme pode observar nas figuras 34, 35 e 36, é possível extrair um arquivo com extensão csv que contem os dados mostrados sobre os orelhões. Entretanto foram feitas várias tentativas sem sucesso para extrair esses dados além do que os aparelhos não vêm com sua geolocalização.

Todas as concessionárias de telefonia devem enviar mensalmente à ANATEL um arquivo contendo o detalhamento de todos os equipamentos de uso público como parte da prestação de contas para o Sistema de Gestão de Metas de Universalização o SGMU.

Por trabalhar na concessionária de telefonia Sercomtel, foi conseguido um exemplo desse arquivo e, de posse dos dados fornecidos em site da ANATEL (http://sistemas.anatel.gov.br/sgmu/Ajuda/Ajuda.asp?SISQSmodulo=14625) sobre o seu layout, foram extraídas as informações.

A princípio o arquivo foi aberto com editor de planilhas Excel, e utilizando suas ferramentas de importação de arquivos com formato csv, foi separado os dados em colunas utilizando como argumento para a separação o ‘;’ (ponto e virgula). Depois foram separadas apenas as colunas relevantes para o sistema. Em seguida foi preciso trabalhar isoladamente com a notação do sistema de posicionamento geográfico, efetuando a conversão para o sistema decimal utilizado pela API do Google. Na sequencia, foi utilizado um poderoso editor de textos para, através da utilização de macros, criar um script de inserção no banco de dados.



Ordem	Descrição do Campo	Observações
1	Número da linha do arquivo	1...N
2	Número do telefone público (TUP)	99999999 Atenção: Utilizar apenas a quantidade de caracteres necessária para fornecer a informação (isto é, não é necessário utilizar "zeros à esquerda") Obs: não insira separador entre os números (traço, ponto, vírgula, etc)
3	Código Nacional - CN (Código DDD)	99
4	Código CNL (identificador de localidade)	99999
5	Data de ativação do TUP	dd/mm/aaaa – ex: 15/01/2006
6	Logradouro	Quantidade máxima de caracteres: 150
7	Número	Quantidade máxima de caracteres: 10
8	Bairro	Quantidade máxima de caracteres: 150
9	Complemento	Quantidade máxima de caracteres: 150
10	CEP	8 caracteres
11	Latitude	**GGDMMSSCC** GG = Grau (0 até 89);

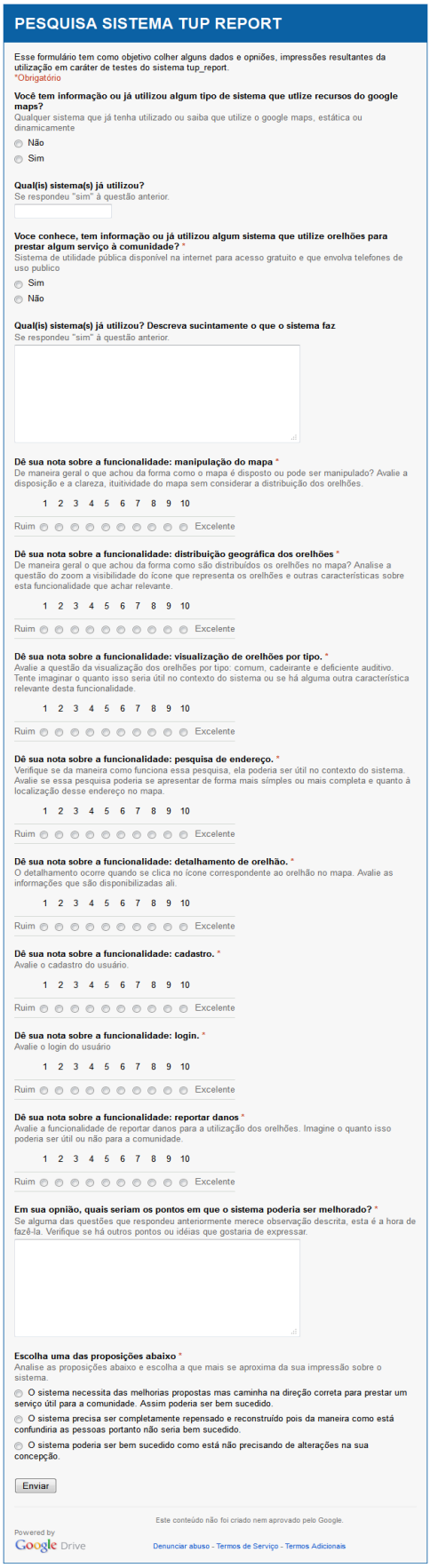
Figura 37 – Figura demonstrando o layout do arquivo de dados sobre os orelhões carregados pelas concessionárias

```
insert into tup_report.tup_localizacao_ (id_tup_localizacao_,tup_lat, tup_lng,
tup_telefone, tup_endereco, tup_bairro, tup_ldn, tup_ldi, tup_acessivel, tup_cadeirante,
tup_deficiente) values (1,-23.3283611111111,-51.2015555555556,'4333996511','VICENTE SUBTIL
ANC, RN° 470','TAMARANA - PR',1,1,1,0,0);
insert into tup_report.tup_localizacao_ (id_tup_localizacao_,tup_lat, tup_lng,
tup_telefone, tup_endereco, tup_bairro, tup_ldn, tup_ldi, tup_acessivel, tup_cadeirante,
tup_deficiente) values (2,-23.2986111111111,-51.1747222222222,'4333996506','DEMETRIO
CARNEIRO SIQUEIRA, RN° 870','TAMARANA',1,1,1,0,0);
insert into tup_report.tup_localizacao_ (id_tup_localizacao_,tup_lat, tup_lng,
tup_telefone, tup_endereco, tup_bairro, tup_ldn, tup_ldi, tup_acessivel, tup_cadeirante,
tup_deficiente) values (3,-23.2613888888889,-51.1755555555556,'4333996516','JERONIMO DA
SERRA, S, RN° 175','TAMARANA',1,1,1,1,0);
```

Figura 38 – Figura demonstrando o script de inserção dos dados na tabela de infromações de orelhão

4.4 TESTES COM OS USUÁRIOS

Foram efetuados testes com alguns usuários que tiveram contato com o sistema por alguns momentos e, logo em seguida responderam um formulário eletrônico como mostra a figura 39.



PESQUISA SISTEMA TUP REPORT

Esse formulário tem como objetivo colher alguns dados e opniões, impressões resultantes da utilização em caráter de testes do sistema tup_report.
*Obrigatório

Você tem informação ou já utilizou algum tipo de sistema que utlize recursos do google maps?
Qualquer sistema que já tenha utilizado ou saiba que utilize o google maps, estática ou dinamicamente

- Não
- Sim

Qual(is) sistema(s) já utilizou?
Se respondeu "sim" à questão anterior.

Voce conhece, tem informação ou já utilizou algum sistema que utilize orelhões para prestar algum serviço à comunidade? *
Sistema de utilidade pública disponível na internet para acesso gratuito e que envolva telefones de uso publico

- Sim
- Não

Qual(is) sistema(s) já utilizou? Descreva sucintamente o que o sistema faz
Se respondeu "sim" à questão anterior.

Dê sua nota sobre a funcionalidade: manipulação do mapa *
De maneira geral o que achou da forma como o mapa é disposto ou pode ser manipulado? Avalie a disposição e a clareza, iuitividade do mapa sem considerar a distribuição dos orelhões.

1 2 3 4 5 6 7 8 9 10
Ruim Excelente

Dê sua nota sobre a funcionalidade: distribuição geográfica dos orelhões *
De maneira geral o que achou da forma como são distribuídos os orelhões no mapa? Analise a questão do zoom a visibilidade do ícone que representa os orelhões e outras características sobre esta funcionalidade que achar relevante.

1 2 3 4 5 6 7 8 9 10
Ruim Excelente

Dê sua nota sobre a funcionalidade: visualização de orelhões por tipo. *
Avalie a questão da visualização dos orelhões por tipo: comum, cadeirante e deficiente auditivo. Tente imaginar o quanto isso seria útil no contexto do sistema ou se há alguma outra característica relevante desta funcionalidade.

1 2 3 4 5 6 7 8 9 10
Ruim Excelente

Dê sua nota sobre a funcionalidade: pesquisa de endereço. *
Verifique se da maneira como funciona essa pesquisa, ela poderia ser útil no contexto do sistema. Avalie se essa pesquisa poderia se apresentar de forma mais símples ou mais completa e quanto à localização desse endereço no mapa.

1 2 3 4 5 6 7 8 9 10
Ruim Excelente

Dê sua nota sobre a funcionalidade: detalhamento de orelhão. *
O detalhamento ocorre quando se clica no ícone correspondente ao orelhão no mapa. Avalie as informações que são disponibilizadas ali.

1 2 3 4 5 6 7 8 9 10
Ruim Excelente

Dê sua nota sobre a funcionalidade: cadastro. *
Avalie o cadastro do usuário.

1 2 3 4 5 6 7 8 9 10
Ruim Excelente

Dê sua nota sobre a funcionalidade: login. *
Avalie o login do usuário

1 2 3 4 5 6 7 8 9 10
Ruim Excelente

Dê sua nota sobre a funcionalidade: reportar danos *
Avalie a funcionalidade de reportar danos para a utilização dos orelhões. Imagine o quanto isso poderia ser útil ou não para a comunidade.

1 2 3 4 5 6 7 8 9 10
Ruim Excelente

Em sua opnião, quais seriam os pontos em que o sistema poderia ser melhorado? *
Se alguma das questões que respondeu anteriormente merece observação descrita, esta é a hora de fazê-la. Verifique se há outros pontos ou idéias que gostaria de expressar.

Escolha uma das proposições abaixo *
Analise as proposições abaixo e escolha a que mais se aproxima da sua impressão sobre o sistema.

- O sistema necessita das melhorias propostas mas caminha na direção correta para prestar um serviço útil para a comunidade. Assim poderia ser bem sucedido.
- O sistema precisa ser completamente repensado e reconstruído pois da maneira como está confundiria as pessoas portanto não seria bem sucedido.
- O sistema poderia ser bem sucedido como está não precisando de alterações na sua concepção.

Enviar

Este conteúdo não foi criado nem aprovado pelo Google.

Powered by Google Drive

Denunciar abuso - Termos de Serviço - Termos Adicionais

Figura 39 – Figura demonstrando o formulário de testes aplicado para os usuários

Os pontos mais importantes levantados pelos usuários haja vista seu perfil (com conhecimento prévio em ambiente de desenvolvimento e utilização de sistemas web) foi conforme encadeado abaixo:

- Ao invés de um cadastro de usuários que teria, entre outras informações, o número de seus documentos, seria interessante integrar o sistema com redes sociais principais como Facebook, como forma de identificar o usuário e reforçar o caráter social do sistema.
- O sistema poderia permitir a visualização no mapa, (quando estivesse em níveis de zoom mais distantes que não gerariam carga de orelhões) informações sobre densidade de aparelhos com problemas, ou outras informações
- O sistema poderia marcar os aparelhos que não estivessem em operação por vandalismo por exemplo.

De maneira geral os testes mostraram que o sistema traça o caminho certo, entretanto é consenso o fato de que ele necessita de uma série de adequações estruturais para entrar em ambiente de produção.

4.5 CONCLUSÃO

Os testes realizados com as funcionalidades apresentadas mostraram que o sistema poderia atender em sua essência os principais propósitos para os quais ele foi desenvolvido que seriam: a visualização da distribuição geográfica dos orelhões e a possibilidade de se enviar informações para a concessionária de que determinado equipamento encontra-se com problemas, ajudando assim a manter a planta de equipamentos da concessionária Sercomtel.

Entretanto para que o sistema se torne viável, seria essencial trabalhar os requisitos não funcionais levantados nesse trabalho em um primeiro momento. Posteriormente seria muito importante implementar as melhorias propostas pelos usuários, principalmente agregando o sistema às principais redes sociais e a implementação das densidades de aparelhos por localidades, população, etc. Bem como uma melhor sinalização dos aparelhos com problemas seria igualmente interessante.

O sistema, produto final das funcionalidades propostas deverá representar a tendência para resolução do problema da falta de informações espaciais sobre a distribuição de telefones públicos das cidades de Londrina e Tamarana. Outras contribuições esperadas são: ajudar a concessionária Sercomtel na manutenção dessa planta, beneficiar a comunidade que poderá consultar aonde estarão disponíveis orelhões comuns e adaptados para pessoas com deficiências.

4.6 TRABALHOS FUTUROS

Como trabalhos futuros pode ser destacado em primeiro plano o desenvolvimento dos requisitos não funcionais (performance, escalabiblidade, acessibilidade, segurança, entre outros) para tornar possível a aplicação em produção desse sistema. Nesse mesmo plano entraria a questão do desenvolvimento de um *layout* mais atrativo e funcional.

Em um segundo plano estaria a implementação do levantado pelos usuários nos testes efetuados, como a integração com redes sociais, marcação dos aparelhos vandalizados, visualização de outras informações de densidade dos equipamentos em menores níveis de *zoom* do mapa.

Em um terceiro plano estaria a utilização de banco de dados de informações geográficas que seria mais adequado para esse tipo de aplicação e o que resultaria em maiores ganhos de performance na utilização desses tipos de sistema.

5 CRONOGRAMA DE EXECUÇÃO

Etapa / Atividade	2012												2013	
	1	2	3	4	5	6	7	8	9	10	11	12	1	2
Revisão Bibliográfica / Estado da Arte	X	X												
Projeto de Interação e Interface			X	X										
Desenho/Implementação/Codificação			X	X	X	X	X	X	X					
Testes com Usuários				X	X	X	X	X						
Análise e Avaliação dos Testes				X	X	X	X	X						
Aplicação de Resultados							X	X						
Novos Testes com Usuários									X					
Análise e Avaliação dos Testes								X	X	X				
Escrita da Monografia					X	X	X	X	X	X	X	X	X	X

REFERÊNCIAS

ANATEL. Agencia Nacional de Telecomunicações. **Telefones Públicos**. Disponível em <http://www.anatel.gov.br/Portal/exibirPortalInternet.do> . Acesso em 01/12/2012

MELO JUNIOR, Cleuton Sampaio de. **Web 2.0 e Mashups**: Reinventando a Internet. Rio de Janeiro: Brasport, 2007.

MERRILL, Duane. **Mashups**: The new breed of Web app. 2006. Disponível em: < http://www.ibm.com/developerworks/web/library/x-mashups/index.html >. Acesso em 25 out. 2012

CÂMARA, Gilberto; DAVIS, Clodoveu; MONTEIRO, Antonio Miguel Vieira; BORGES, Karla. **Introdução à Ciência da Geoinformação.** Disponível em: <http://www.dpi.inpe.br/gilberto/livro/introd/index.html> Acesso em 22 out. 2012

BEZERRA, Eduardo. **Princípios de análise e projetos de sistemas com UML**: Elsevier, 2007

GOOGLE. **API JavaScript do Google Maps v3**. Disponível em: <https://developers.google.com/maps/documentation/javascript/?hl=pt-br > Acessado em 13 ago. 2012

ORACLE. **Oracle Application Server Containers for J2EE 10g (9.0.4)**:Frequently Asked Questions. Disponível em: <http://www.oracle.com/technetwork/middleware/ias/oc4j-faq-jsp-904-090451.html> Acessado em 10 jan. 2013

W3C. Consórcio World Wide Web.**Visão geral do HTML5**: Capitulo 1. Disponível em: <http://www.w3c.br/cursos/html5/conteudo/capitulo1.html> Acessado em: 10 jan 2013

W3C. Consórcio World Wide Web.**Web Style Sheets Home Page**. Disponível em: <http http://www.w3.org/Style/> Acessado em: 05 jan 2013.

ECMA. **ECMA-262:**ECMAScript Language Especification. Disponível em: <http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf> Acessado em: 11 jan 2013

JQUERY. **What is JQuery.** Disponível em: <http://jquery.com/> Acessado em: 11 jan. 2013

GLASSFISH. **Get Started** Disponível em < http://glassfish.java.net/public/getstarted.html> Acesssado em 15 jan 2013