CENTRO UNIVERSITÁRIO FEEVALE

ISMAEL LIZAKOSKI

DESENVOLVIMENTO DE PROTÓTIPO LBS WEBMOBILE COM USO DA API J2ME

Novo Hamburgo, novembro de 2008.

ISMAEL LIZAKOSKI

DESENVOLVIMENTO DE PROTÓTIPO LBS WEBMOBILE COM USO DA API J2ME

Centro Universitário Feevale
Instituto de Ciências Exatas e Tecnológicas
Curso de Ciência da Computação
Trabalho de Conclusão de Curso

Professor Orientador: Juliano Varella de Carvalho

Novo Hamburgo, novembro de 2008.

AGRADECIMENTOS

Ao meu orientador Juliano Varella de Carvalho pela motivação, pela disponibilidade e dedicada orientação; a paciência e compreensão dos meus pais Vanderlei Ricardo Lizakoski e Isabel Cristina Vetter Lizakoski, ao meu irmão Rafael Lizakokis e minha cunhada Raquel Homem; meu amigo Rubens Plentz Gonçalves pela amizade, disponibilidade de horários nas correções e dicas.

Aos colegas formandos e professores pelas palavras amigas de incentivo e motivação. Enfim, a todos que de algumas formas contribuíram para a conclusão desse trabalho, meu muito obrigado.

"Cultura para mim é:

Fazer parte do presente, com alma

No passado e os olhos no futuro."

(Clodovil)

RESUMO

Location-Based Services (LBS) é um conjunto de serviços que usa informações geográficas, combinadas ou não com a posição do terminal móvel, para obter e gerar informações entre os usuários de dispositivos móveis. LBS permite que uma aplicação seja executada em dispositivos móveis, com base na plataforma *Java 2 Micro Edition* (J2ME), onde há possibilidade de realizar consultas via web a dados georeferenciados, coletados de um *Global Positioning System* (GPS). A J2ME é ideal para esse padrão de aplicação LBS, fornecendo um conjunto de classes e funcionalidades voltadas para esse fim e dispõe de outras características como: entrega dinâmica de conteúdo; segurança; compatibilidade entre plataformas; conteúdo interativo; acesso *offline*; o poder da linguagem orientada a objetos e a padronização da indústria através do *Java Community Process*. Este trabalho tem como objetivo desenvolver um protótipo de software para telefone celular que permita ao usuário fazer consultas via WEB a uma base de dados georeferenciada, além de estudar conceitos acerca de GPS, banco de dados geográficos e *webmapping*.

Palavras-chave: LBS (serviços baseados em localização), *API Location – JSR-179,* J2ME, GPS e banco de dados geográficos.

ABSTRACT

Location-Based Services (LBS) is a set of services which uses geographical information, combined or not with the position of the movable terminal, to get and generate information among the users of movable devices. LBS allows an application to be accomplished, based on the platform Java 2 Micro Edition (J2ME), when there is the possibility of carrying out consultations to geo-referenced data through the web, which are collected from a Global Positioning System (GPS). J2ME is ideal for this application pattern in LBS, providing a set of classes and functionalities geared towards this intention and counts on other characteristics such as: dynamic content delivery; security; compatibility between platforms; interactive content; offline access; the power of language directed to objects and to the standardization of industry through Java Community Process. This paper aims to develop a software prototype for cell phones which allows the user to carry out consultations to a geo-referenced data base through the WEB, besides studying the concepts about GPS, geographical data bank and web mapping.

Keywords: LBS (services based on location), API Location – JSR-179, J2ME, GPS and geographical data bank.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A-GPS	*Assisted-GPS*
AOA	*Angle of Arrival*
API	*Application Programming Interface*
CDC	*Connected Device Configuration*
Cell ID	*Cell Identification*
CLDC	*Connected Limited Device Configuration*
DM	Dispositivos móveis
FP	*Foundation Profile*
GPS	*Global Positioning System*
GSM	Sistema Móvel Global
IMP	*Information Module Profile*
KVM	Máquina Virtual K
JCP	*Java Community Process*
J2EE	*Java 2 Entrerprise Edition*
J2ME	*Java 2 Micro Edition*
J2SE	*Java 2 Standard Editon*
JSR	*Java Specification Request*
JSR-179	*Location API for J2ME*
JVM	Máquina Virtual Java
LBS	*Location-Based Services*
OGC	*OpenLS*
MIDP	*Mobile Information Device Profile*
NAVSTAR	*Navegation Satellite with Time and Ranging*
NMEA	*National Marine Eletronics Association*
PBP	*Personal Basis Profile*

PP	*Personal Profile*
PPS	*Precision Positioning Service*
SIG	Sistemas de informação geográfica
SPS	*Standard Positioning Service*
TDOA	*Time Differencial of Arrival*
TIMATION	*Time Navigation*

SUMÁRIO

1 INTRODUCAO

Este trabalho tem como objetivo desenvolver um protótipo de software para telefone celular que permita ao usuário fazer consultas via WEB a uma base de dados georeferenciada, além de estudar conceitos acerca de GPS, banco de dados geográficos e *webmapping*.

Sistemas ou serviços baseados em localização *Location-Based Services* (LBSs) podem ser vistos como a convergência de várias tecnologias atuais, tais como:

- Sistemas de comunicação móvel;
- Tecnologias de localização;
- Dispositivos móveis (DM) com a Internet;
- Sistemas de informação geográfica (SIG);
- Servidores da aplicação, banco de dados geográficos e *webmapping*.

O ponto central desses sistemas está em obter informações que determinem a localização de dispositivos móveis (DM) e, baseado nessa informação, oferecer serviços de acordo com o contexto de utilização do sistema. É objetivo, através deste trabalho, apresentar a *API Location* da tecnologia *J2ME (JAVA 2 Micro Edition)*, utilizada para implementação de sistemas *LBS* com suporte a *GPS (handset-based)*. Mostrar seu funcionamento para gerar serviços georeferenciados, suas principais classes e especificações.

O celular permite ao usuário acesso a serviços independentemente de sua localização, com suporte a mobilidade e existência de software de comunicações sem fio ou GPS. Diversos serviços podem ser implementados para telefone celular, acesso correio eletrônico, entre outros.

Dois exemplos para elucidar: o médico pode consultar a ficha do paciente no hospital pelo seu computador de mão; o turista pode consultar a base de dados do serviço de turismo da cidade que ele está visitando. (JOHNSON, 2007)

Exemplos de utilização de dispositivos portáteis podem ser notados em diversos setores e com as mais variadas finalidades. Aplicações que disponibilizam aos vendedores acesso, a qualquer momento, à tabela de preços atualizada, disponibilidade e outras informações referentes a um produto, por meio de um aparelho celular, com possibilidades inclusive de efetuar o pedido do cliente em tempo real; softwares que possibilitam o acesso móvel à informações corporativas de uma empresa, como visto em (MIRANDA; MARCONDES, 2004), são alguns exemplos de aplicações.

Vindo ao encontro destas tecnologias, a plataforma *Java 2 Micro Edition* (J2ME), caso específico do ambiente Java, está se tornando a solução preferida pela grande maioria dos fabricantes de dispositivos para telefonia celular. Um reflexo disso é a quantidade de celulares que já dispõem da Máquina Virtual Java (JVM) instalada. Além destes aparelhos, outros dispositivos *wireless* ou GPS como telefone celular também dão suporte a J2ME.

Acompanhando a evolução dos serviços baseados em localização e da crescente oportunidade de capitalizar no mercado, empresas de Tecnologia da Informação, como a *Sun Microsystems* que criou a plataforma J2ME e a linguagem de programação Java, procura disponibilizar interfaces de programação de aplicativos (APIs) que capacitem programadores a desenvolver aplicações para dispositivos móveis capazes de obter, manipular, armazenar e distribuir a sua localização. A API disponibilizada pela Sun denomina-se "*Location*" ou "JSR-179" (JAVADOC, 2006).

Portanto, o trabalho apresenta princípios da arquitetura J2ME e o funcionamento da *API Location*. Apresenta também, conceitos de GIS, serviços baseados em localização (LBS) e métodos de obtenção de dados de posicionamento, e por fim, envolve conteúdos do universo *Location-Based Services* (LBS) para desenvolver um protótipo de software para telefone celular que permita ao usuário fazer consultas via WEB a uma base de dados georeferenciada, além de estudar outros conceitos tais como GPS, banco de dados geográficos e *webmapping*.

Este trabalho está dividido em cinco capítulos, estando assim distribuídos:

No primeiro capítulo, encontra-se a introdução e os objetivos a serem alcançados com o desenvolvimento do trabalho.

No segundo capítulo é apresentado um estudo sobre *Location-Based Services* (LBS). Incluindo ainda algumas seções referentes aos serviços baseados em localização.

O terceiro capítulo trata de SIG.

O quarto capítulo trata da tecnologia Java. Apresenta ainda um estudo mais aprofundado na plataforma J2ME.

O quinto capítulo trata de uma pesquisa sobre telefonia celular, onde será apresentado um exemplo de uma aplicação utilizando J2ME que utiliza princípios de localização, para encontrar pontos de referência com base em cálculos geográficos.

O último capítulo trata das considerações finais sobre este projeto, sugestões e idéias para trabalhos futuros.

2 LBS WEBMOBILE

Existem diversas definições para os LBS, do inglês *Location Based Services* (serviços baseados em localização). A empresa (WHEREONEARTH, 2006) apresenta uma definição sintética, mas abrangente desse conceito: “LBS podem ser descritos como aplicações que reagem a uma causa geográfica”. A organização OpenGIS[1] apresenta a seguinte definição: “Os serviços de localização são todos os tipos de software de serviços que acedem, fornecem ou reagem a uma localização geográfica”. Os LBS permitem aos usuários de dispositivos móveis utilizarem serviços baseados em sua posição ou localização geográfica (ROCHA, 2001).

Com efeito, os LBS não são mais do que serviços baseados na localização atual do usuário. A iniciativa do serviço prestado pode partir do usuário, como por exemplo, saber a localização de um dado tipo de restaurante, e o modo de como chegar lá; ou então pode partir do servidor LBS, por exemplo, ao detectar a proximidade do usuário de um dado restaurante

[1]OpenGIS (ou OGIS) é um padrão de interoperabilidade entre SIGs, baseado em conceitos de orientação por objetos, que estabelece mecanismos normatizados para acesso a informações geográficas. OpenGIS Consortium. Technical Specifications. www.opengis.org/techno/specs.htm. Acesso em: 11 de Março de 2008.

que lançará uma promoção, o LBS gera de imediato uma mensagem ao usuário. (BAÇÃO; COSTA, 2006).

O LBS é, basicamente, a convergência de várias tecnologias atuais, tais como: comunicação móvel, tecnologias de localização, dispositivos móveis (DM) com Internet, sistemas de informação geográfica (SIG), servidores de aplicações e banco de dados (Oracle, MySQL, etc.). Em resumo, pode ser definido como qualquer serviço com valor agregado, cuja principal função é obter informações que determinem a localização do dispositivo móvel e, baseado nas mesmas, oferecer serviços de acordo com o contexto de utilização, a partir de vários métodos de indexação e serviços de orientação.

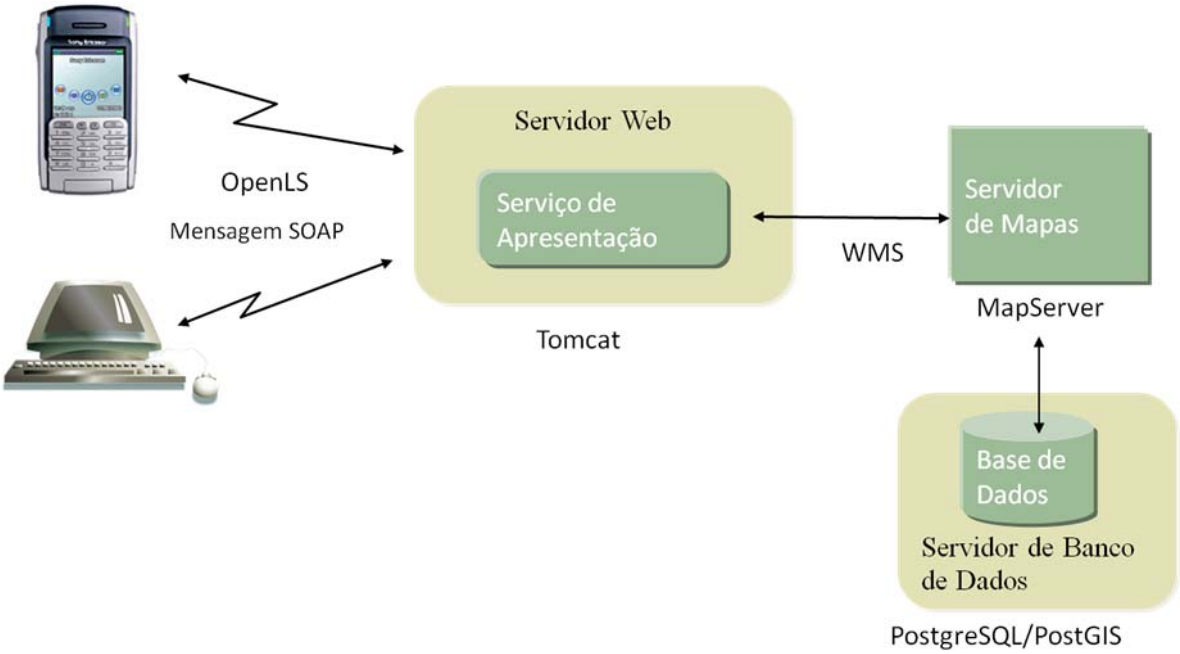

FIGURA 2.1 – Exemplo de Arquitetura LBS

2.1 Tecnologias de localização de dispositivos móveis

Serviços baseados em localização (LBS) são usados para definir e prover serviços baseados na posição geográfica de um dispositivo móvel. As tecnologias de localização (ou posicionamento) informam a localização de um dispositivo móvel. As tecnologias utilizadas para a localização podem ser classificadas quanto ao local onde as coordenadas de posicionamento são responsáveis por oferecer facilidades ao usuário, geralmente em incursões a localidades não conhecidas anteriormente, e situações imprevistas.

Normalmente, um LBS pode fornecer informações geo-personalizadas, ajudando as pessoas na localização de prestadores de serviços (borracheiro, postos de gasolina,

restaurantes, hospitais, supermercado, mercado financeiro, casas noturnas, etc.), encontrando o melhor caminho até o destino (menor distância e tráfego), providenciando socorro em casos de emergência, gerenciando frotas, rastreando cargas e recuperando veículos roubados, entre outros serviços.

A plataforma de Serviços Baseados em Localização – LBS (*Location Based Services*) oferece às operadoras e seus clientes, provedores de serviço e portais, uma variedade de serviços que combinam o perfil pessoal e a localização de cada usuário.

Fornecer serviços GPS baseados em localização é um desafio complexo do ponto de vista da tecnologia e dos negócios. É necessário que haja a integração em tempo real de vários elementos-chave. Esses elementos são: determinação da localização, dispositivo de mapeamento, conteúdo e informação, aplicações, personalização e uma interface otimizada com o dispositivo do usuário final. As operadoras de telefonia sem fio estão buscando um modo simples e de baixo custo para integrar esses elementos e introduzir serviços de localização.

A área de entretenimento oferece serviços de bate-papos, programação de cinema/teatros, músicas e jogos interativos multiusuários. Aplicações colaborativas oferecem mais uma oportunidade de aumento de produtividade, reduzindo custos e prazos. Vários tipos de interação entre usuários e empresa poderiam ser realizados durante o horário de trabalho, porém em ambiente fora da empresa.

Um operador de seguros poderia realizar maior quantidade de sinistros num período, tendo acesso a dados e documentos corporativos no momento e local do acidente. Ao término da vistoria, os documentos estariam prontos a serem enviados à matriz e o funcionário estaria liberado para outro serviço.

Serviços de comércio eletrônico oferecem comodidade e rapidez ao cliente na compra de produtos, no acesso a bancos e em outros serviços disponibilizados ao cliente (SHARMA, 2001 apud TAFFE, 2002). Aplicações de telemetria tais como alarmes contra roubo e incêndio, máquinas de venda automática, estatísticas de tráfego de veículos e medições (água, eletricidade e gás), ainda são pouco exploradas.

Existe uma quantidade enorme de serviços e conteúdos voltados ao usuário de computadores fixos. Estas aplicações não são facilmente acessíveis a usuários de dispositivos

computacionais móveis, fazendo com que surjam alternativas como aplicações móveis via voz. Nestas aplicações, as entradas e/ou saídas são feitas através da fala ao invés de uma interface gráfica. O aplicativo pode estar localizado no dispositivo ou em algum servidor na Internet (LOUREIRO, 2001).

Na prática, aplicações e serviços poderiam estar integrados: serviços baseados em localização estariam associados aos de comércio eletrônico móvel e poderiam ser solicitados através de uma interface de voz.

A miniaturização dos componentes eletrônicos permitiu o desenvolvimento de computadores que cabem em um celular ou *palm* de mão. Esses computadores caracterizam-se por apresentar várias limitações quanto ao poder de processamento, capacidade de armazenamento, autonomia da bateria, tamanho da tela e meios de entrada de dados. Apesar de todos esses obstáculos, sua mobilidade e portabilidade superam suas desvantagens.

Atualmente, a tecnologia *wireless* atinge um estágio tão avançado de desenvolvimento, que está se tornando um dos tópicos mais instigantes da atualidade. Isso porque ela tem o poder de derrubar as fronteiras entre comunicação, informação, mídias e entretenimento, oferecendo todos estes serviços no celular de mão, em qualquer lugar e a qualquer hora (ROCHA, 2001). Embora tudo isso ainda seja algo novo para muitas pessoas, as empresas envolvidas com esta tecnologia já estão implantando vários serviços para dispositivos *wireless*, dentre eles os *Location Based Services*.

Os métodos de localização podem ser divididos em duas principais categorias: as baseadas em redes e as baseadas em *Handset* (ou seja, o dispositivo é responsável por descobrir a localização). Os métodos baseados em redes vão desde métodos relativamente simples, como o *Cell–ID*, a métodos mais complexos como o *Time Difference of Arrival* (TDOA). Os métodos baseados em *Handset* fornecem a localização sem a ajuda de redes de comunicação móveis como GSM (Sistema Móvel Global). O GPS é um dos mais conhecidos métodos dessa categoria. Existem também métodos híbridos que unem recursos dos métodos baseados em redes e em *Handset*, o A-GPS é um deles, onde tenta-se diminuir o tempo de demora na inicialização do receptor de GPS usando a informação vinda da rede celular.

2.2 Método Baseado em *Handset*

2.2.1 Global Positioning System

Em 1973, o Departamento de Defesa dos Estados Unidos proveu a fusão de dois projetos, TIMATION (*Time Navigation*) da Marinha e 621B da Força Aérea para a concepção de um novo sistema denominado NAVSTAR/GPS. O *Navegation Satellite with Time and Ranging* (NAVSTAR) ou *Global Position System* foi projetado para fornecer a posição instantânea bem como a velocidade de um ponto sobre a superfície da Terra ou próximo a ela, num referencial tridimensional (JÚNIOR, 2003). Como o GPS foi desenhado por razões militares, foram criados dois tipos de serviços de localização:

- PPS (*Precision Positioning Service*), restrito somente para uso governamental.

- SPS[2] (*Standard Positioning Service*), disponibilizado para uso comercial.

Este sistema é baseado em satélites que enviam mensagens específicas que são interpretadas pelo receptor GPS.

O GPS é dividido em três segmentos principais:

– segmento espacial, constituído pelos satélites;

– segmento de controle, constituído pelas estações terrestres que controlam o desempenho e o funcionamento do sistema;

– segmento usuário, constituído pelos usuários do sistema.

[2] SPS é livre para usuários civis, usando receptores GPS comerciais disponíveis. Os militares americanos, intencionalmente, deterioraram a precisão horizontal, vertical e de tempo para uso comercial com a *Selective Availability Rule*. Mas uma decisão do Presidente Bill Clinton em remover a *Selective Availability Rule* promete dar total apoio para uso comercial das informações de localização.

Em sua estrutura final o sistema conta com 24 satélites espalhados por 6 órbitas diferentes. Esta configuração faz com que qualquer ponto na superfície da Terra, ou próxima a ela, tenha a cobertura de no mínimo 4 satélites. A figura 2.2 demonstra a estrutura do sistema GPS (JÚNIOR, 2003).

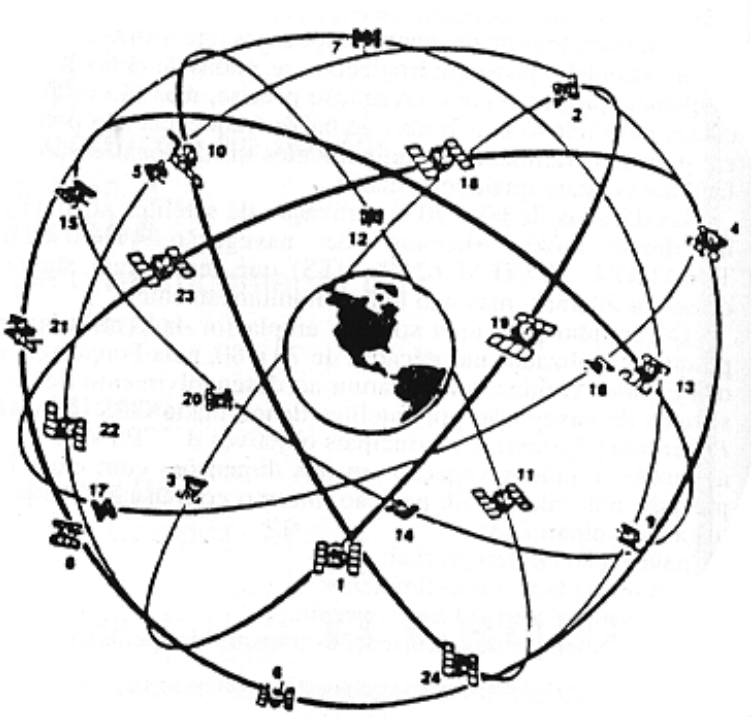

FIGURA 2.2 - Conjunto de Satélites utilizado no GPS. Fonte: Júnior, 2003.

As aplicações dos receptores GPS variam em precisão e funcionalidade. Alguns possuem programas que fazem transformações entre sistemas de coordenadas ou contém dados de saída compatíveis com sistemas de SIG comuns no mercado. Outros dispositivos permitem a leitura com os receptores em movimento e são bastante úteis para realizar mapeamento de terrenos com veículos (CÂMARA, 1996).

As vantagens de se utilizar receptores GPS são a precisão da informação e o tamanho destes dispositivos, que podem ser encontrados até mesmo em relógios de pulso. A desvantagem do uso destes receptores é que eles não conseguem ter muita precisão em ambientes fechados (*indoor*) (ROCHA, 2001).

Atualmente, o desenvolvimento e aperfeiçoamento de novas tecnologias de localização para uso *indoor* estão contribuindo ainda mais para o surgimento de aplicações que integram informação de localização com serviços. Além disso, empresas passaram a oferecer serviços altamente personalizados para seus usuários baseados não apenas em sua

localização, mas também no seu perfil e outras informações de contexto, como, por exemplo, tempo/hora do dia.

2.3 Método Híbrido

2.3.1 Assisted-GPS

Segundo (ZURSTRASSEN 2003) a idéia do *Assisted-GPS* (A-GPS) é distribuir as tarefas de posicionamento entre os dispositivos móveis e o servidor de serviços de localização. O servidor se comunica com os satélites da rede mapeando sua região de controle e passa estas informações para o dispositivo móvel. Desta forma este dispositivo tem necessidade de fazer todos os cálculos de sua localização sozinhos, pois recebe estas informações do servidor pré-processadas.

Os principais benefícios do A-GPS são a redução do tempo de localização, economia de energia e simplificação dos requisitos do receptor de GPS. A desvantagem é que este método é mais caro em relação ao GPS.

Ele utiliza informações da rede para determinar mais rapidamente a posição dos quatro satélites a serem “ouvidos”. A célula da rede distribui a localização desses satélites e pode drasticamente reduzir o tempo de inicialização que receptores GPS precisam. Além disso, esse método economiza bateria já que o receptor GPS somente é acionado em áreas úteis.

Um tecnologia que utiliza esta tecnologia é o *SnapTrack’s Location on Demand* (QUALCOMM, 2003). Ele é basicamente uma tecnologia de localização com marketing de precisão. Caso o usuário opte por não ser localizado, o serviço é suspenso e ainda permite que o consumo da bateria seja reduzido (ZURSTRASSEN, 2003). A figura 2.3 demonstra um exemplo de uma aplicação baseada nesta tecnologia.

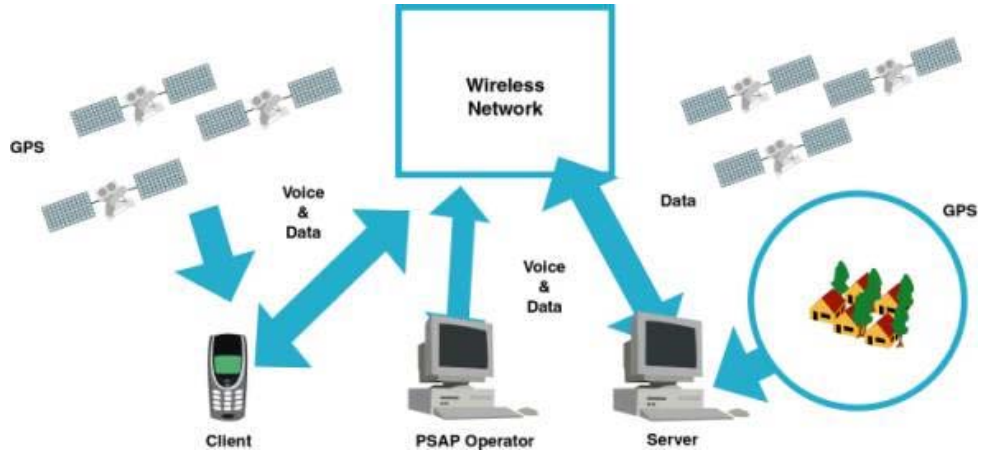

FIGURA 2.3 - Tecnologia *Assited-GPS*. Fonte: ZURSTRASSEN, 2003.

Essa solução vem se tornando a mais comum nas redes celulares, por causa de sua qualidade de fornecer a informação de localização em lugares fechados ou com condições mais severas como áreas urbanas com grandes prédios ou locais com muitas árvores. Mas o mais importante é que A-GPS oferece a informação sobre a posição de uma forma mais rápida que o GPS padrão, pois usa as células da rede para distribuir a informação da localização dos satélites, isso reduz bastante o tempo de inicialização em relação a um terminal GPS simples, além de também consumir menos energia.

2.4 Métodos baseados na rede

2.4.1 Cell Identification

A *Cell Identification* (Cell ID) é uma tecnologia onde a localização é determinada através da célula onde o usuário se encontra. *Cell ID* ou localização baseada em células se dá da seguinte maneira: um servidor armazena as informações de cada célula, latitude e longitude da posição da torre, ângulo inicial e final dos setores da estação e o raio máximo de cobertura. Com posse destas informações o servidor poderá calcular o centróide de cada setor (ZURSTRASSEN, 2003).

Todas estas informações servirão para o cálculo da localização dos assinantes quando esta for solicitada ao servidor de aplicações LBS. Com isto, informações referentes ao posicionamento do usuário serão diretamente proporcionais ao tamanho da célula onde se encontram. Sendo que esta variação poderá ser de 150m a até 20Km. Embora esta seja a grande desvantagem desta tecnologia, o fato de a localização ser muito rápida e de fácil disponibilização ainda a torna uma boa opção quando se fala em soluções *netwok-based*. A figura 2.4 exemplifica o uso da tecnologia *Cell Identification.*

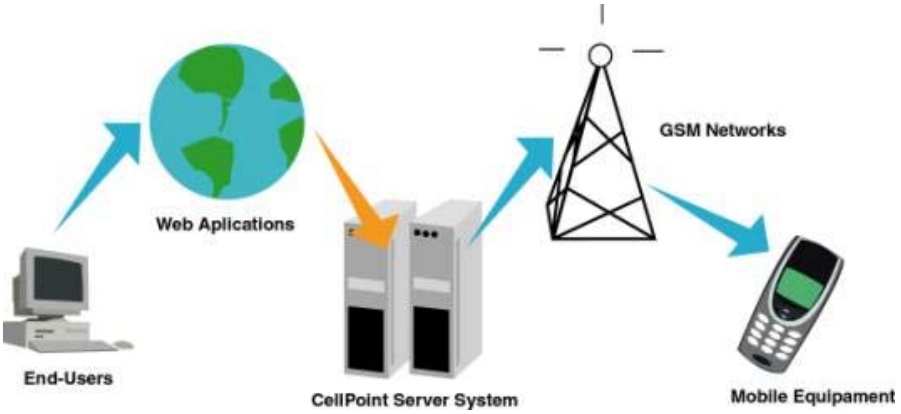

FIGURA 2.4 - Tecnologia *Cell Identification* - Fonte: ZURSTRASSEN, 2003.

É predominantemente usada nas redes GSM, é considerado um serviço de nível básico. Esse sistema identifica a célula na qual o usuário está conectado, e então mapeia isso em coordenadas, fazendo uma estimativa de localização do usuário. Como em sistemas 3G são usadas células pequenas, essa informação sobre a posição dos usuários nessas redes acaba tendo uma grande precisão. Uma das vantagens desse sistema é que ele funciona perfeitamente em ambientes fechados (onde, por exemplo, satélites GPS teriam dificuldades de obter uma localização).

2.4.2 Angle of Arrival

O *Angle of Arrival* (AOA) é uma das mais conhecidas abordagens ao problema da localização. Seu princípio está no cálculo do ângulo com que determinado sinal chega às antenas. O AOA é determinado pela variação de fase dos sinais recebidos ao longo de um grupo de antenas. A diferença de fase do sinal entre antenas deste grupo resulta em ângulos recebidos, e isto pode ser referenciado em relação a qualquer direção fixa. O conhecimento do AOA de pelo menos dois locais, fornece a informação necessária para a localização da fonte do sinal (SILVA, 2003).

Esta tecnologia é muito menos dispendiosa que a do GPS, por aproveitar os sinais transmitidos pelos aparelhos móveis, dispensando a atualização quer dos aparelhos, quer da rede de comunicações. Uma das desvantagens desta tecnologia é que a precisão da localização é degradada sempre que existir interferência no sinal (ROCHA, 2001). A figura 2.5 exemplifica o uso da tecnologia AOA.

FIGURA 2.5 - Tecnologia *Angle of Arrival*. Fonte: ZURSTRASSEN, 2003.

Esse método usa um equipamento especial que tem que ser instalado nas estações base para determinar o ângulo de chegada para os sinais de rádio. Com alguns cálculos geométricos, se pode determinar a localização do usuário com somente duas estações base recebendo o sinal.

2.4.3 Time Difference of Arrival

O *Time Differencial of Arrival* (TDOA) usa receptores de rádio freqüência que são instalados em vários lugares da antena para capturar sinais. Nesta abordagem, a posição do dispositivo móvel é calculada pela medição das diferenças de tempo do sinal recebido em cada receptor (ROCHA, 2001).

A precisão deste tipo de tecnologia é determinada pelas limitações de largura de banda do sinal, bem como a precisão do sistema e pelo ambiente em que se propaga. Outras desvantagens incluem a extrema precisão temporal necessária em cada receptor, a necessidade de recalibrar estes mesmos receptores devido a diferentes influências no sinal. A vantagem desta tecnologia é o fato de não ser necessário modificar os dispositivos móveis (SILVA, 2003). A figura 2.6 demonstra o uso da tecnologia TDOA.

FIGURA 2.6 - Tecnologia *Time Differencial of Arrival*. Fonte: ZURSTRASSEN, 2003.

Esse método usa a diferença de chegada dos sinais de rádio até as estações base (a partir do terminal móvel). É necessário pelo menos três estações base para calcular a localização do usuário.

2.5 OpenGIS Location Services

A especificação *OpenLS* (OGC, 2004) foi aprovada pelo consórcio *OpenGIS* em Janeiro de 2004. Ela define um conjunto de interfaces para o desenvolvimento de serviços baseados em localização, todos utilizando protocolos padrão Web. Os serviços especificados encontram-se descritos a seguir:

- Serviço de Diretório: provê acesso a um diretório *online* para localização de um determinado lugar, produto ou serviço.

- Serviço de *Gateway*: identifica a posição geográfica de um determinado dispositivo móvel.

- Serviço de Geocodificação reversa: identifica uma posição geográfica dado o nome de um lugar ou endereço. Também funciona de forma reversa identificando um endereço completo dada uma posição geográfica.

- Serviço de Apresentação de Mapas: apresenta informações geográficas no terminal móvel. É usado para apresentar mapas destacando rotas entre dois pontos, pontos de interesse, áreas de interesse, localizações e/ou endereços.

– Serviço de Determinação de Rotas: determina a rota entre dois pontos informados pelo usuário. O usuário também pode, opcionalmente, informar pontos pelos quais a rota deve passar, rotas preferenciais (mais rápida, mais curta, menos tráfego, mais atrativa, etc.) e o modo de transporte.

2.6 Banco de Dados Geográficos

O *PostGIS*, utilizando neste trabalho, é um módulo de extensão espacial para o SGBDOR *PostgreSQL* que permite armazenar, recuperar e analisar dados espaciais, freqüentemente utilizados em um SIG. Foi desenvolvido pela *Refractions Research Inc*[3], como uma tecnologia de banco de dados espacial e vem sendo mantida pela mesma (POSTGIS, 2005).

2.6.1 Objetos Espaciais

Segundo Silberschatz et al (1999, p.700), bancos de dados geográficos são banco de dados espaciais utilizados para armazenar informações geográficas, como mapas. Já Câmara et al (2005, p.2-3), utiliza o termo sistemas gerenciadores de dados geográficos, como um mecanismo dentro de um SGBD que serve para armazenar dados e também a geometria dos objetos espaciais.

A partir da utilização de banco de dados geográficos, é possível construir aplicações para manipulação de dados espaciais de forma mais coesa e ainda usufruir das vantagens que possui um SGBD. Dentre elas, cita-se a persistência dos dados, integridade, segurança e um

[3]A Refractions é uma companhia de consultoria em GIS e banco de dados localizada em Victoria, British Columbia, Canadá, especializada em integração de dados e desenvolvimento de software. (POSTGIS, 2005).

aumento da eficiência, pois com o uso de um SGBD, as informações podem ser extraídas e alteradas de modo mais prático e eficaz (SILBERSCHATZ et al., 1999).

Os objetos espaciais que o *PostGIS* presta suporte são as feições[4] definidas pelo consórcio *OpenGIS*. Atualmente, ele suporta as características simples e a representação de APIs[5], mas não os operadores de comparação e convolução[6] definidos pela especificação de implementação de feições para SQL do *OpenGIS* (POSTGIS, 2005).

Há três formas de trabalhar com dados geográficos. A primeira é a utilização da estratégia Dual, onde utiliza-se um SGBD como repositório dos atributos convencionais dos objetos geográficos e arquivos em separado para armazenar as representações geométricas destes (CÂMARA, 1995).

A segunda opção é a utilização de um SGBD Relacional (SGBDR) para armazenar os dados espaciais, tanto seus componentes espaciais e alfanuméricos. Outra opção é a utilização de extensões espaciais desenvolvidas sobre SGBD Objeto – Relacional (SGBDOR), que definem funcionalidades e procedimentos capazes de armazenar, acessar e analisar dados espaciais (FERREIRA, 2003).

Utilizando-se de extensões espaciais pode-se usufruir de tipos de dados espaciais definidos por elas, tais como ponto, linha e polígono. Estas extensões permitem que tais dados sejam manipulados como qualquer outro tipo de dado de SGBD. Além desta característica, há

---

[4] Feições são uma representação abstrata de fenômenos do mundo real, associada a uma localização relativa a terra. (BOGORNY, 1999).
[5] API: Application Programming Interface. Interface entre o computador e os programas do usuário.
[6] A convolução é uma técnica de processamento de imagens.

a extensão da linguagem SQL para proceder as consultas sobre estes dados, ofertando operações e funções para consultar relações espaciais (SANTOS, 2006).

2.7 Aplicações de LBS

Atualmente, uma das áreas mais interessantes no uso dos LBS é a de assistência em emergências. De tal maneira que nos EUA, o *Federal Trade Commission* (FTC) elaborou uma lei denominada E911Act destinada a padronizar e aprimorar o serviço de emergência 911 através de dispositivos móveis. Esta lei está em vigor desde 1999 e seu objetivo é melhorar o tempo de resposta nas chamadas de emergência feitas por usuários de telefones celulares (DEITEL, 2003).

No Brasil, já é possível encontrar aplicativos LBS baseados em soluções *handset*. Um deles é o Apontador *Duo* (WEBRASKA, 2003), um guia de ruas com recursos de voz e GPS. Este software foi desenvolvido para um dispositivo do tipo *Palm Pilot*, onde são armazenados os mapas de acordo com a rota determinada pelo usuário. Outra área que terá um grande impulso a partir do uso de serviços de localização será a do comércio eletrônico móvel (*m-business*). Com a combinação entre os dispositivos móveis e os LBS, o *m-business* terá como elaborar estratégias de *marketing* extremamente poderosas.

Pode-se fazer com que uma pessoa que tenha um aparelho celular receba uma mensagem informando que o restaurante que está a 500m de distância fará uma promoção relâmpago em meia-hora.

Aplicações LBS serão a próxima “grande onda” no mundo do desenvolvimento GPS (FALEIROS, 2006) e a *Java 2 Micro Edition* é ideal para tornar-se um padrão para o desenvolvimento desse tipo de aplicação, pois fornece um conjunto de classes e funcionalidades voltadas para esse fim, e dispõe de outras importantes características como (FALEIROS, 2006):

- Entrega dinâmica de conteúdo;
- Segurança;
- Compatibilidade entre plataformas;
- Conteúdo interativo;

– Acesso *Offline*;

– O poder da linguagem orientada a objetos;

– Padronização da indústria através do *Java Community Process*.

Aplicações que usam LBS são limitadas apenas pela imaginação dos desenvolvedores, mas existem algumas categorias onde necessidades e oportunidades foram claramente identificadas:

– Mapeamento e navegação são alvos óbvios para LBS, mas desenvolvedores que combinem esta capacidade com serviços de direção e outras ofertas serão os vencedores a longo prazo.

– Rastreamento de força de trabalho e aplicações de gerenciamento podem acabar sendo as aplicações mais lucrativas porque ajudarão as empresas a otimizar a utilização de pessoas, suprimentos e equipamento. Outros segmentos incluem segurança, entrega e serviços de negócios.

– Entretenimento e jogos são áreas que podem aproveitar o posicionamento geográfico e criar jogos e serviços interativos de alto nível.

– Apliçações "*Finder*" podem utilizar a localização do usuário para ajudar a encontrar qualquer coisa, desde o banheiro mais próximo até um amigo numa multidão.

– Aplicações multimídia podem se aproveitar das informações de localização para rotular os dados enquanto eles são salvos.

– Aplicações metereológicas podem prover o usuário com previsões dinâmicas sobre os locais por onde ele passa.

– Aplicações de lembretes baseadas em localização podem alertar os usuários quando eles passarem por certos lugares, mostrando, por exemplo, uma lista de compras quando um usuário passar por um supermercado, ou uma consulta marcada, ao se aproximar do consultório de seu médico.

Diversas formas de cobrança estão disponíveis, dependendo do método de posicionamento e da operadora. Alguns métodos, como GPS, são gratuitos, enquanto outros são disponíveis sob requisição ou taxas de admissão.

3 SISTEMA DE INFORMAÇÃO GEOGRÁFICA

Um Sistema de Informação Geográfica, em inglês, *Geographic Information System* (GIS), é um sistema para criação e gerenciamento de dados espaciais e atributos associados descritivos. No sentido exato, é um sistema de computador capaz de integrar, armazenar, editar, analisar, compartilhar, e exibir informação geograficamente referenciada. Em um sentido mais genérico, GIS é uma ferramenta de mapas inteligentes que permite aos usuários criarem consultas interativas (pesquisas criadas pelo usuário), e então analisarem a informação espacial e editarem os dados. (GEOGRAPHIC, 2006).

Sistemas de Informações Geográficas são modelos do mundo real úteis a um certo propósito que subsidiam o processo de observação (atividades de definição, mensuração e classificação), a atuação (atividades de operação, manutenção, gerenciamento, construção, etc.) e a análise do mundo real. (GEOGRAPHIC, 2006)

Estes sistemas são constituídos por uma série de programas e processos de análise, cuja característica principal é focalizar o relacionamento de determinado fenômeno da realidade com sua localização espacial. Eles utilizam uma base de dados computadorizada que contém informação espacial, sobre a qual atuam uma série de operadores espaciais e baseiam-se numa tecnologia de armazenamento, análise e tratamento de dados espaciais, não-espaciais e temporais e na geração de informações correlatas. (MENEZES; MEDEIROS, 2005)

Um GIS é muitas vezes associado aos mapas. No entanto, mapa é apenas um meio pra se trabalhar com dados geográficos em um GIS, e apenas um tipo de produto gerado por ele. Um GIS pode ser visto de três formas (GIS, 2006):

- Visão de banco de dados (geodatabase) - GIS é baseado numa estrutura de base de dados que descreve o mundo em termos geográficos;

- Visão de mapas (geovisualização) - GIS é um conjunto de mapas inteligentes e outras visões que mostram características e relacionamentos entre características na superfície da Terra. Mapas no nível de informações geográficas podem ser construídos e utilizados como “janelas na base de dados” para suportar consultas, análises, e edição de informação;

- Visão de modelos (geoprocessamento) - GIS é um conjunto de ferramentas de transformação de informações que deriva novos conjuntos de dados geográficos a partir de um conjunto de dados existentes. Essas funções de geoprocessamento pegam informações de um conjunto de dados existentes, aplicam funções analíticas e escrevem resultados em novos conjuntos de dados derivados.

A tecnologia GIS possui uma série de padrões estabelecidos pelo *Open Geospatial Consortium*[7] (OGC), que foi fundado em 1994 por várias organizações da área a fim de ditar padrões para a tecnologia da informação geográfica.

3.1 Estrutura de um SIG

Um SIG pode ser dividido em componentes ou subsistemas (CAMARA, 1996), conforme o esquema mostrado na figura 3.1:

[7] Open Geospatial Consortium (OGC, 2005a) é um consórcio com mais de 250 companhias, agências governamentais e universidades, criado para promover o desenvolvimento de tecnologias que facilitem a interoperabilidade entre sistemas envolvendo informação espacial e localização (Gardels, 1996) (Percivall, 2003). Os produtos do trabalho do OGC são apresentados sob forma de especificações de interfaces e padrões de intercâmbio de dados.

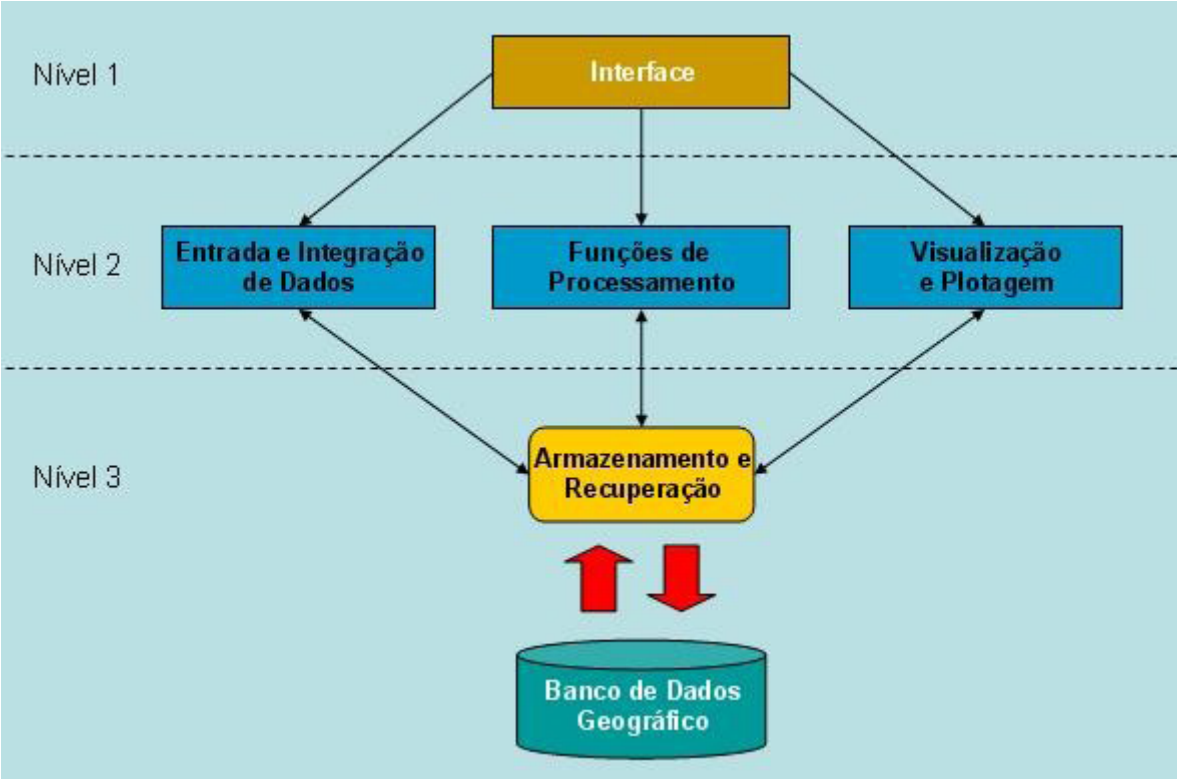

FIGURA 3.1 - Componentes de um SIG. Fonte: CAMARA, 1996.

- Nível 1: a interface com o usuário contempla as diretrizes de controle e operacionalização do sistema, sendo este o nível que está mais próximo do usuário final;

- Nível 2: correspondente à entrada e integração de dados, funções de processamento, visualização e plotagem, é o nível responsável pelo processamento dos dados de entrada e saída, possibilitando a edição, análise e visualização de dados;

- Nível 3: permite o gerenciamento da base de dados geográficos, o armazenamento e a recuperação de dados espaciais, sendo o nível mais interno do sistema.

Para que um SIG possa armazenar dados geográficos é necessário levar em consideração que o banco de dados tenha condições de tratar os diversos componentes da informação geográfica (CASANOVA, 2005): atributo, espaço e tempo. Esses componentes permitem, respectivamente, responder questões como “o quê?”, “onde?” e “quando?”, embora o aspecto temporal não seja ainda bem explorado nos SIG.

Um sistema de gerência de banco de dados geográficos (SGBDG) é um componente fundamental de um SIG, responsável por armazenar, manipular e recuperar os tipos de dados geográficos. Os SGBDG devem garantir que as propriedades fundamentais de SGBD convencionais sejam aplicáveis a dados geográficos.

Estas propriedades incluem três requisitos importantes (CAMARA 1994):

- Eficiência: acesso e modificação de grandes volumes de dados;
- Integridade: controle de acesso por múltiplos usuários;
- Persistência: manutenção de dados por longo tempo, independentemente dos aplicativos que acessam o dado.

Os dados convencionais procuram descrever algumas características que existem nos objetos espaciais.

Por exemplo, no cadastro pertencente a uma cidade atributos como nome da rua, número do lote, nome do proprietário, são exemplos. Por outro lado, os dados espaciais são caracterizados pela localização geográfica sobre a superfície terrestre em certo instante, e estão associados a variáveis como latitude, longitude etc, e são armazenados com base em um sistema de coordenadas. Dados deste tipo são modelados no SIG para representar polígonos, linhas, pontos ou objetos complexos, como por exemplo, uma rede de esgoto ou malha rodoviária.

Tanto a documentação quanto a administração dos dados passam pelo uso de metadados, que podem ser definidos como uma abstração dos dados, ou ainda, dados de um nível mais alto que descrevem dados de um nível inferior. Os metadados ficam armazenados no catálogo do SGBD (ELMASRI, 2004), e apresentam uma descrição concisa da estrutura do banco de dados e suas restrições, e é a partir daí que as informações serão processadas, atualizadas e consultadas. Caso não seja estabelecida uma estratégia para administração da documentação do banco de dados, isso poderá trazer com o decorrer do tempo, problemas de inconsistência à base.

As operações de consulta e manipulação dos dados georeferenciados são, em essência, as duas grandes funções de um SIG. São operações que permitem extrair

informações e realizar simulações do mundo real, buscando entender seu comportamento e até mesmo prever situações futuras.

4 J2ME

Em 1991 a *Sun Microsystems*, diante da concepção de que a próxima área em que os microprocessadores teriam um impacto profundo seria a de dispositivos eletrônicos inteligentes destinados ao consumidor final, financiou uma pesquisa para o estabelecimento de uma linguagem capaz de atender os requisitos destes dispositivos. Esta pesquisa tinha o codinome Green e o seu resultado foi o desenvolvimento de uma linguagem baseada em C e C++ que seu criador, James Gosling, chamou de *Oak* (carvalho) em homenagem a uma árvore que dava para a janela de seu escritório na Sun. Mais tarde descobriu-se que já havia uma linguagem de computador com este nome. Foi quando uma equipe da Sun que estava numa cafeteria local sugeriu o nome Java, que é uma cidade de onde um tipo específico de café é importado (DEITEL, 2001).

Mas como o mercado naquela época não se desenvolveu tão rápido como era esperado, e neste mesmo período a *World Wide Web* (WWW) estava tendo um grande impulso, a Sun vislumbrou o potencial de utilizar Java para criar páginas da *Web* com o chamado conteúdo dinâmico. Foi desta forma que Java ganhou força e vem se consolidando como uma das linguagens mais conhecidas e utilizadas mundialmente (DEITEL, 2003).

Atualmente Java está distribuída em três plataformas: *Java 2 Entrerprise Edition* (J2EE), *Java 2 Standard Editon* (J2SE) e *Java 2 Micro Edition* (J2ME). Estas três plataformas formam um conjunto que possibilita o desenvolvimento de aplicações completas, robustas e portáveis.

A J2SE foi a primeira edição criada da *Java 2 Plataform* . Esta plataforma dá suporte para o desenvolvimento de aplicações para desktop e estações de trabalho. J2SE é a versão básica do Java com *Applications Programming Interface* (API) padrão (DEPINÉ, 2002).

Segundo SOUJAVA (2003), J2EE é um conjunto de especificações coordenadas, objetivando o desenvolvimento de aplicações corporativas. As especificações do J2EE são criadas por um grupo muito grande de empresas, entre elas *Sun*, *IBM*, *BEA*, *Oracle*, *Borland*, *Apache*, e para as quais existem dezenas de produtos compatíveis hoje no mercado.

A mais recente edição é a J2ME. Esta tecnologia é direcionada ao mercado de pequenos dispositivos. Seu conjunto de especificações tem por objetivo disponibilizar uma JVM, API e ferramentas para equipamentos como: telefones celulares, *pagers*, *handhelds*, vídeo games, sistemas embutidos, etc (SOUJAVA, 2003).

Os telefones celulares que possuem câmera, por exemplo, geralmente utilizam Java para gerenciamento do dispositivo. Jogos com capacidade de reconhecimento dos movimentos humanos, também integram essa vasta lista de aplicativos J2ME.

As figuras 4.1 e 4.2 representam as três edições de Java, bem como os dispositivos para os quais cada camada foi implementada.

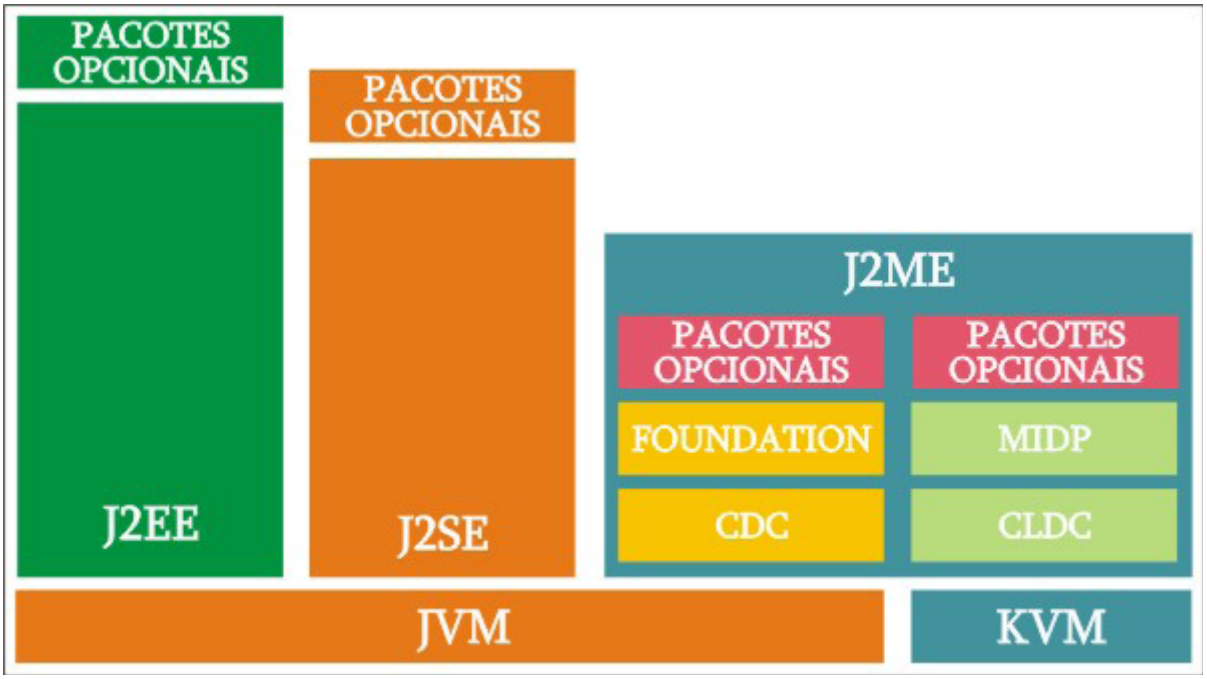

FIGURA 4.1 - Arquitetura da plataforma JAVA. Fonte: MUCHOW, 2004.

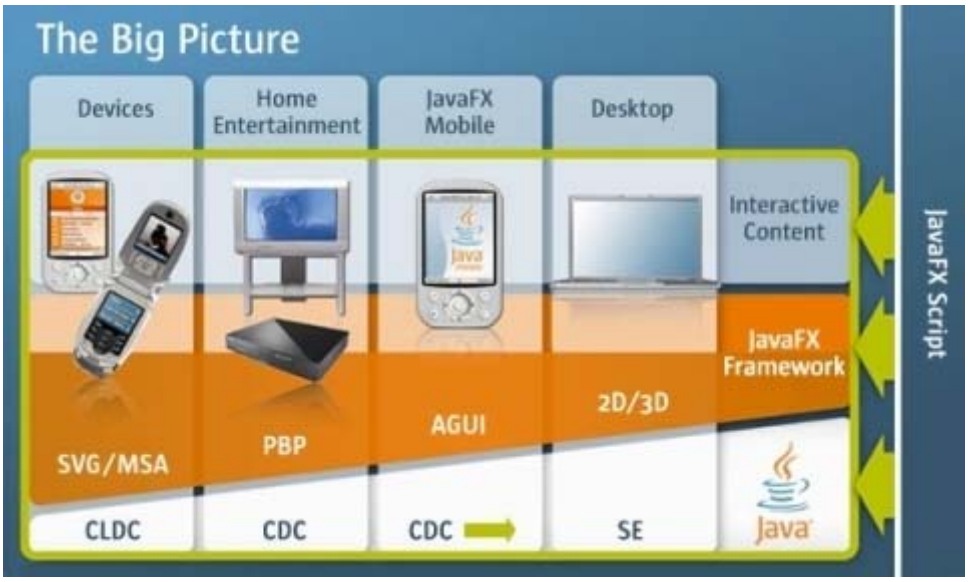

FIGURA 4.2 - Arquitetura JAVA. Fonte: MUCHOW, 2004.

A figura 4.1 ilustra as diferentes camadas que compõem a tecnologia Java. Sua arquitetura compreende sistemas em diversas plataformas e dispositivos diferentes. A figura 4.2 mostra a especificação J2ME (*JavaFX*), e as suas aplicabilidades.

Os principais fatores que tornam Java uma linguagem de destaque são (ORACLE, 2002):

- Orientação a Objetos: a programação orientada a objetos é um paradigma natural e poderoso usado no desenvolvimento de aplicativos. Pesquisas apontam que o uso deste paradigma permite uma sobrevivência mais prolongada às mudanças que acompanham o crescimento de um negócio;

- Portabilidade e independência de plataforma: programas *Java* são interpretados por uma máquina nativa *(Java Virtual Machine* - JVM) em tempo de execução. Pelo fato destes programas executarem debaixo do controle da JVM, os programas Java podem ser utilizados em qualquer sistema operacional que tenha uma JVM;

- Facilidade de distribuição: classes Java podem ser baixadas dinamicamente na internet quando requeridas. Adicionalmente, *Java* provê suporte para cliente-servidor e programação distribuída;

- *Multithreading*: a linguagem *Java* disponibiliza primitivas de concorrência para o programador. Isto significa que é possível desenvolver aplicativos que executam múltiplas tarefas em paralelo;

- Robustez e segurança: *java* tem a capacidade interna de impedir a corrupção de memória. Automaticamente administra os processos de alocação de memória e a checagem de limite de *array*.

4.1 Java 2 Micro Edition

O J2ME consiste em uma coleção de APIs (*Application Programming Interface*) ou Interfaces de Programação de Aplicativos da linguagem Java, e a especificação de uma máquina virtual, que visa possibilitar a programação para dispositivos móveis com maior flexibilidade e portabilidade.

A intenção é que um programa seja feito uma única vez, em uma única linguagem, mas que possa ser executado em diversos dispositivos (MUCHOW, 2004). No entanto, levando-se em conta a enorme quantidade de aparelhos móveis existentes, onde cada um possui suas próprias características de processamento, quantidade de memória disponível, conectividade, consumo de energia, entre outros aspectos, o J2ME adota o mesmo conceito de máquina virtual visto nas plataformas *Java Standard* (J2SE) e *Enterprise* (J2EE), aumentando assim a portabilidade das aplicações. A arquitetura da plataforma J2ME dentro do ambiente Java é mostrada nas Figuras 4.3 e 4.4.

Na arquitetura do J2ME a camada entre o sistema operacional e o hardware do dispositivo é a máquina virtual, que define quais limitações e recursos que o dispositivo atende.

Assim para executar um aplicativo J2ME, um dispositivo deve conter uma implementação da maquina virtual definida para a sua categoria de dispositivos (HEMPHILL; WHITE, 2002). A Figura 4.3 expõe a arquitetura geral da plataforma J2ME.

FIGURA 4.3 - Plataforma J2ME. Fonte: MUCHOW, 2004.

Para aumentar a eficiência da plataforma, acima da máquina virtual existem ainda os níveis de configuração e perfil, e cada aparelho é encaixado em uma configuração e perfil, de acordo com o escopo de suas características. Porém, por não ser uma definição exata, alguns dispositivos podem ser incluídos em mais de uma categoria.

4.1.1 Configuração na Plataforma *Java 2 Micro Edition*

Uma configuração está vinculada diretamente à máquina virtual Java, possível de ser utilizada no dispositivo e define as funcionalidades e serviços que devem ser oferecidos por ela ao programador. As configurações definem as APIs básicas, baseadas nas características essenciais do dispositivo.

Atualmente são definidas duas configurações:

- *Connected Device Configuration* (CDC): configuração destinada a dispositivos que possuem maior poder computacional, ou seja, maior processamento, maior quantidade de memória, etc. Exemplos de dispositivos desta categoria são: *set-top boxes* para televisores, alguns PDAs, sistemas de navegação para automóveis, etc. A seguir algumas características típicas dos dispositivos dentro dessa configuração (MICROSYSTEMS, 2006):

– 512 kilobytes no mínimo de memória para execução do *Java*;

– 256 kilobytes pelo menos, para alocação de memória em tempo de execução;

– Conectividade de rede, largura de banda possivelmente persistente e alta.

- *Connected Limited Device Configuration* (CLDC): designada para dispositivos menores, mais lentos e com maiores limitações, com capacidade de conexão em rede.

Nessa categoria se encaixam aparelhos celulares, *pagers* e PDAs. Os requisitos de hardware para dispositivos dessa categoria segundo (MICROSYSTEMS, 2003) são:

– 160 kilobytes de memória não-volátil (que preserva o conteúdo mesmo com o dispositivo desligado), para a máquina virtual e as bibliotecas CLDC;

– 32 kilobytes de memória volátil, disponível durante o tempo de execução da aplicação.

Mesmo sendo uma divisão clara, conforme a evolução tecnológica contínua venha oferecer maior possibilidade, com aumento de processamento, memória e outros recursos, a sobreposição entre as categorias irá se tornar maior, aumentando assim o significado e necessidade de existência dos perfis (MUCHOW, 2004).

4.1.2 Perfil na Plataforma *Java 2 Micro Edition*

Um perfil pode ser tratado como uma extensão da configuração. Uma configuração pode conter vários perfis que por sua vez são especificados para apenas uma configuração. Assim, para aumentar a especificação dos dispositivos com relação às suas características, o J2ME adota esse conceito, onde um perfil é um conjunto de bibliotecas de mais alto nível, tratando de características mais específicas dos dispositivos (MUCHOW, 2004) (HEMPHILL; WHITE, 2002). Os perfis atualmente definidos são (CARDOSO, 2006) (PINHEIRO, 2003):

– Perfis definidos para CLDC:

– *Mobile Information Device Profile* (MIDP): que já conta com duas versões.

– *Information Module Profile* (IMP): é baseado no MIDP, é destinado a dispositivos com interface mais limitada.

– Perfis definidos para CDC:

– *Foundation Profile* (FP): nível mais baixo de perfil para configuração CDC. Fornece implementação de rede e é utilizado principalmente para construção de aplicações sem uso de interface.

– *Personal Profile* (PP): perfil CDC usado em dispositivos que precisam de suporte completo de interface. Usado em PDAs e consoles de jogos. Possui suporte para compatibilidade total com a AWT e o modelo de aplicação *applet*.

– *Personal Basis Profile* (PBP): é uma divisão do PP, fornece um ambiente para dispositivos conectados e com suporte básico de apresentação gráfica, suporte a modelo de aplicação xlet8 (é semelhante a *applets*, as aplicações são descarregadas dinamicamente e executadas).

A utilização de um perfil é opcional, porém é de extenso valor, pois fornecerá recursos para que sejam desenvolvidas aplicações para tipos de dispositivos específicos de forma customizada. Tratando de características relativas à espécie de dispositivo alvo, tornando o resultado da aplicação mais eficaz.

4.1.3 APIs opcionais na plataforma J2ME

As APIs opcionais são funcionalidades adicionais específicas que não serão encontradas em todos os dispositivos de uma determinada configuração ou perfil, mas importantes o suficiente para serem padronizadas. Criadas para disponibilizar requisitos muito específicos de aplicações, esses pacotes opcionais oferecem APIs padrão para uso em ambas tecnologias existentes e emergentes como conectividade à base de dados, mensagens sem fio, multimídia, gráficos 3D, *Web Services* e Localização. Pelo fato dos pacotes opcionais serem modulares, fabricantes de dispositivos evitam sobrecarregar os dispositivos de funcionalidades desnecessárias instalando apenas os pacotes que as aplicações realmente irão utilizar. Os pacotes opcionais podem ser implementados lado a lado virtualmente a qualquer combinação de configurações e perfis.

As especificações da plataforma J2ME, assim como também ocorre para as edições J2SE e J2EE da plataforma Java, são desenvolvidas sob o amparo da *Java Community Process* (JCP). Uma especificação Java inicia sua vida como uma requisição de especificação, *Java Specification Request* (JSR). Um grupo experiente que consiste de representantes de companhias participantes, é formado para criar a

especificação. A JSR passa então por diversos estágios no JCP antes de ser finalizada. Toda JSR recebe um número. Especificações J2ME são comumente referenciadas pelas suas JSRs. Por exemplo, a JSR179 corresponde a *API Location* do J2ME.

No modelo de solução apresentado nesse trabalho utilizaremos algumas API opcionais além de outros recursos do J2ME. Outras APIs do J2ME de menor relevância utilizada no modelo e outras APIs possíveis de serem utilizadas não serão discriminadas.

4.1.4 Máquina Virtual Java

Na plataforma J2ME as configurações definem os requisitos que a maquina virtual do dispositivo deve atender, tendo em vista que todo aplicativo *Java* precisa ser executado em uma máquina virtual. Para os aplicativos pertencentes à configuração CDC, a especificação da maquina virtual é a mesma do J2SE. Já na configuração CLDC, foi desenvolvida uma especificação de maquina virtual que considera as restrições de recursos dos dispositivos que se encaixam nesta categoria. A implementação referência da maquina virtual criada pela *Sun Microsystems* chama-se KVM[8] (maquina virtual K), mas não é necessariamente a única que pode estar executando em um dispositivo CLDC (MUCHOW, 2004).

4.2 *API Location* com *Java Specification Request* (JSR-179)

A *API Location* para J2ME define um pacote opcional (*javax.microedition.location*) com o objetivo de possibilitar que desenvolvedores de

[8] KVM é uma máquina virtual compacta e portável, designada para dispositivos com grandes restrições de memória, tais como telefones sem fio, *pagers*, PDAs, entre outros.

software possam construir aplicativos ou serviços baseados em localização (LBS – *Location Based Service*) para dispositivos móveis com recursos limitados, como telefones celulares, e que possam ser implementados com qualquer método de localização comum (MAHMOUD, 2005). Então, a sua proposta é de ser compacta e genérica produzindo informações sobre localização física para aplicações Java.

Essa API surgiu a partir da JSR-179 liderada por um desenvolvedor da Nokia Corporation, Kimmo Loytana, e um grupo de empresas ligadas ao ramo de telefonia móvel (LOYTANA, 2005). Ela foi aceita pelo Comitê Executivo para Micro Edição do *Java Community Process* (JCP) no mês de junho de 2003. Esse ambiente demonstra a importância dessas JSR e a preocupação e empenho por parte das empresas no desenvolvimento de uma API de localização.

Para que esse pacote Java possa ser utilizado é exigido como configuração J2ME do dispositivo a *Connected Device Configuration* (CDC) ou a *Connected Limited Device Configuration* (CLDC) versão 1.1, pois suportam números em ponto flutuante necessários na armazenagem dos valores de latitude e longitude. Por outro lado, ela não depende de um perfil J2ME em particular, podendo ser MIDP ou "*Personal Profile*". No entanto, recomenda-se a utilização do perfil MIDP 2.0 por possuir conceitos renovados de segurança, já que obter informações de localização são peculiares e devem ser controladas e restritas para certas aplicações. Os três principais recursos que a *API Location* traz para o desenvolvedor são:

1. Obter informação sobre a localização de um dispositivo;

2. A possibilidade de criar, editar, armazenar e recuperar pontos da superfície;

3. A possibilidade de obter a orientação de um dispositivo.

A plataforma de hardware é quem determina quais os métodos de localização são suportados. A API necessita se conectar a um método provedor de localização, o qual gera localizações.

Métodos provedores de localização diferem um dos outros de diversas maneiras. Nesse sentido, uma descrição das principais abordagens e tecnologias disponíveis para localização é apresentada nos capítulos 2 e 3 – LBS e GIS. Por

exemplo, o uso de alguns métodos pode custar mais do que outros, e precisões suportadas por provedores variam individualmente. Além dos aspectos tecnológicos, aspectos financeiros também podem ser aplicados, visto que alguns métodos podem ser de graça, enquanto outros podem depender de serviços pagos (NOKIA-a, 2006).

Cabe à aplicação a responsabilidade de definir critérios de seleção do método de localização. É baseada nos critérios definidos e no hardware disponível que a API escolhe a melhor forma de atender à requisição da aplicação. Os critérios suportados pela *API Location* podem ser: precisão, tempo de resposta, necessidade por altitude e velocidade.

Esses critérios podem ainda ser divididos em fortes e fracos (HAIGES, 2005):

– Critérios fortes envolvem características como utilizar o custo máximo por consulta, definir se há necessidade de informação de velocidade e definir se endereços são importantes ou coordenadas são suficientes.

– Critérios fracos definem precisão das coordenadas horizontal e vertical no processo de posicionamento.

4.2.1 Ambiente de desenvolvimento

As aplicações baseadas na *API Location* podem ser desenvolvidas e testadas em ambiente simulado. Por exemplo, a ferramenta *Nokia Prototype SDK 3.0* pode ser utilizada para simular a rota de um local até outro numa aplicação MIDlet de localização. Para testar as aplicações, valores de localização podem ser gerados através da ferramenta “Route” inserida no SDK 3.0. Através dessa ferramenta o testador pode desenhar uma rota e gravá-la no formato NMEA (*National Marine Eletronics Association*) 0183 para uso pela aplicação. O formato NMEA 0183 é um padrão comumente utilizado na transmissão de dados GPS (KLIEMANN, 2006).

4.2.2 Utilização da *API Location*

A *API Location* é definida no pacote *javax.microedition.location* do J2ME. Todas as classes da API estão dentro do mesmo pacote. Para descobrir se a API está disponível em um dispositivo utiliza-se a propriedade de sistema

microedition.location.version. O valor contido nessa propriedade do sistema é a versão implementada da especificação da API, por exemplo, “1.0”.

Tabela 4.1: Classes necessárias para escolher o provedor de localização

Classe	*Definição*
Criteria	Utilizado para seleção do provedor de localização.
LocationProvider	Representa a fonte de informação de localização (ex: módulo GPS).

Fonte: NOKIA, 2006

Tabela 4.2: Ouvintes da *API Location*

Classe	*Definição*
LocationListener	Ouvinte que recebe eventos associados com um *LocationProvider* particular.
ProximityListener	Ouvinte de eventos associados com a detecção de aproximação de coordenadas registradas.

Fonte: NOKIA, 2006

A aplicação baseada em localização solicita uma instância da classe *LocationProvider* utilizando como parâmetro uma instância da classe *Criteria* com os critérios a serem satisfeitos pelo provedor de localização. Se obtiver sucesso, esse objeto retornado poderá ser utilizado para obter dados de localização de duas maneiras:

- Invocar um método sincronicamente para obter uma única localização (classe *Location*). Esse método é bloqueante, pois suspende a execução da aplicação até obter resposta ou até expirar o *timeout* - tempo limite especificado em parâmetro (*Location getLocation(int timeout)*); ou

- Registrar um ouvinte e obter atualizações em intervalos definidos pela aplicação. Para isso a aplicação deve implementar as interfaces *LocationListener* e *ProximityListener* (KLIEMANN, 2006).

Tabela 4.3: Classes necessárias para medição da localização

Classe	*Definição*
Location	Representa o conjunto padrão de informação de localização (coordenadas com marcação de tempo, velocidade, precisão, curso, etc.)
AddressInfo	Armazena informação de endereço textual sobre a localização.
Coordinates	Representa valores de coordenadas como latitude, longitude e altitude.
LocationException	É lançada quando um erro específico da API ocorre.
QualifiedCoordinates	Representa valores de coordenadas como latitude, longitude e altitude que são associados com um valor de precisão.

Fonte: NOKIA, 2006

A classe *Location* é uma abstração para representar os resultados de localização. O objeto retornado da requisição contém todos os dados como coordenadas, tempo, etc. As coordenadas são representadas por duas classes:

- O objeto *Coordinates* representa os pontos de latitude e longitude em graus, e altitude em metros; e

- O objeto *QualifiedCoordinates* contém latitude, longitude e altitude, e, além disso, indica a sua precisão, representada como o raio de uma área (KLIEMANN, 2006).

Tabela 4.4: Classes de marcação e armazenamento de pontos de terreno

Classe	*Definição*
Landmark	Representa uma localização conhecida com um nome.
LandmarkException	É lançada quando um erro é relatado no manuseio das *Landmarks*.
LandmarkStore	Métodos para gerenciamento de gravação persistente de *Landmarks*. Grava, apaga e retorna *Landmarks*.

Fonte: NOKIA, 2006

Segundo a especificação da API, são definidos alguns recursos disponíveis por ela como obrigatórios e outros como opcionais, como segue abaixo (JAVADOC, 2008):

1. Recursos obrigatórios para todos os métodos de localização são:

– Prover coordenadas de latitude e longitude e sua precisão;

– Prover marcação de tempo do momento da medição;

– Suportar *LandmarkStore* para armazenar marcas de terreno (*landmark*).

2. Recursos cuja disponibilidade depende do método utilizado:

– Prover informação de altitude;

– Prover precisão de altitude;

– Prover informação de velocidade e de curso;

– Prover informação textual de endereços relacionado à localização; e

– Prover marcas de terreno (*landmark*) de eventos na proximidade.

3. Recursos opcionais cuja disponibilidade depende da implementação de arquivamento de marcas de terreno (*landmark*) no terminal e sua possível relação de compartilhamento de *landmark* com aplicações nativas, são:

– Criar e apagar registros de *landmark*;

– Adicionar e remover categorias de *landmark*.

4. Dependendo do hardware do terminal, outros recursos podem ser obtidos:

– Prover ângulo horizontal de circunferência da orientação do terminal;

– Prover informação de inclinação e rotação 3D da orientação do terminal.

No caso de segurança das informações do usuário, o perfil MIDP (*Mobile Information Device Profile*) versão 2.0 oferece um *framework* para isto, com uma série de vantagens. Ele pode, por exemplo, restringir o acesso a dados de localização exigindo do usuário uma confirmação de permissão explícita, entre outros.

Na figura abaixo, é apresentado o diagrama de classes da *API Location*:

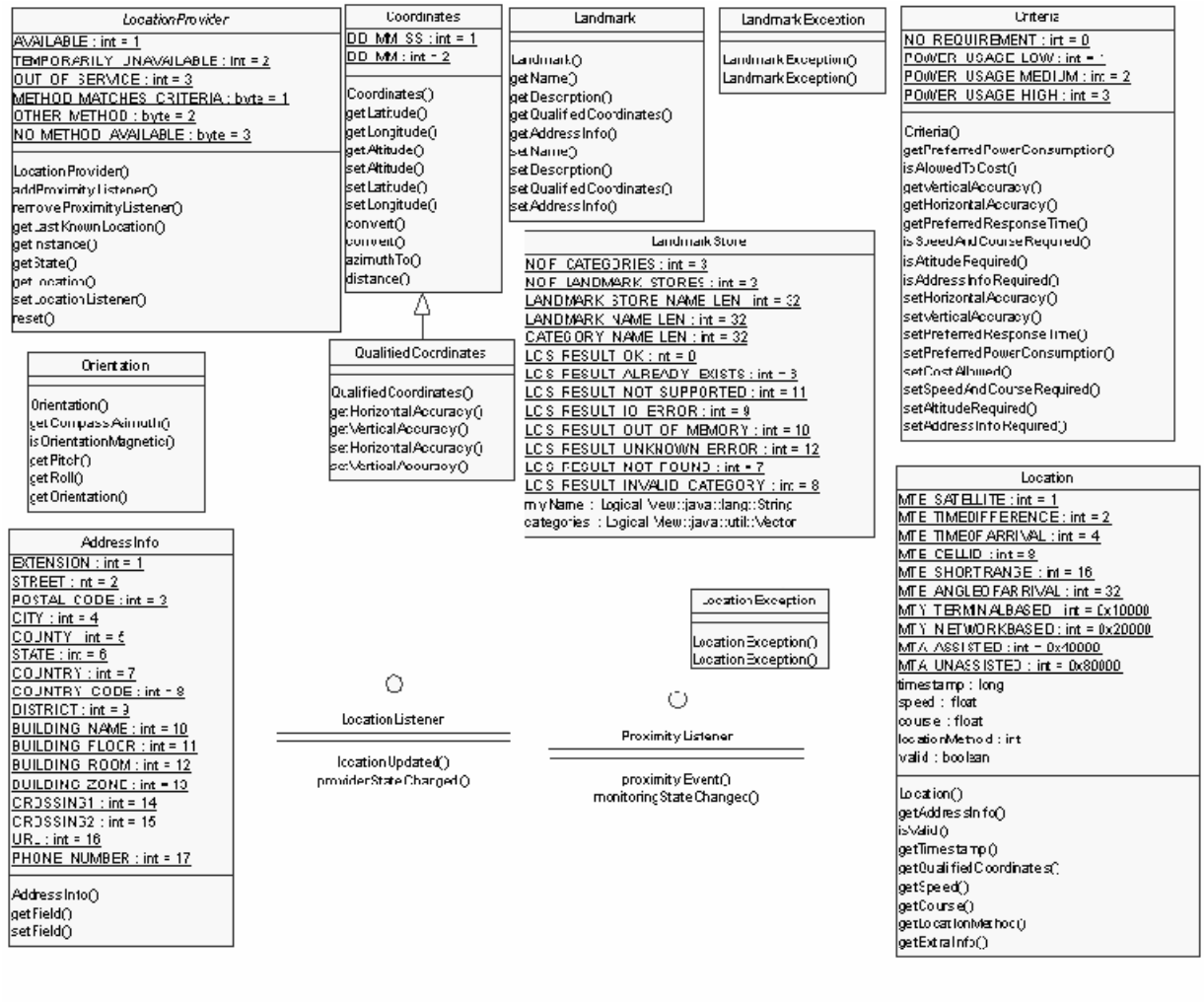

FIGURA 4.4 - Diagrama de Classes da *API Location* (FALEIROS, 2006)

4.2.3 Dispositivos que suportam a *API Location* (JSR-179)

Atualmente, ainda são poucos os dispositivos que já vêm com a *API Location* disponível. No entanto, como muitas APIs J2ME, a tendência é que elas tornem-se presentes em todos os celulares, fazendo parte da futura versão do que hoje temos do JTWI do *Java*. JTWI é a especificação que define a plataforma padrão de indústria para a próxima geração de telefones celulares com tecnologia baseada em *Java*. Ela define os requisitos mínimos e características de tempo de execução que todos os dispositivos sem fio deverão suportar.

Listas com dispositivos que suportam a *API Location* podem ser encontradas na Internet através do site do Java ME (2006) e do site J2ME Polish (2006). Alguns modelos de acordo com o fabricante são:

– BlackBerry: 7520

– Motorola: i730, i88s;

– Nokia: 6265, 6265i, 9500, 5300, 6100, 3100, N95, N93i, 6681, 6600, 770, N800, N810;

– Siemens: C65, C66, CX65, C72, C75, CF75, CX75, M65, S65, S66, SK65, SL65, SP65, S6C, S6V.

4.3 Exemplos de aplicações utilizando J2ME (Sistema proposto)

A aplicação que foi desenvolvida utilizou tecnologia Java (J2ME) e roda em um dispositivo móvel (celular).

O software permite ao usuário efetuar consultas via Web a uma base de dados georeferenciada, funcionando como um localizador geográfico de objetos. Por exemplo, ao ser efetuada a entrega de determinado produto, o entregador imediatamente pode solicitar ao software as informações sobre um local de abastecimento (posto de gasolina) mais próximo, podendo então ter o controle da quantidade de combustível que deve utilizar para chegar em determinado destino, e se necessário determinar o tempo que precisa para encontrar um posto mais próximo.

Unindo suas principais funcionalidades, o sistema de uma maneira geral permitirá localizar um posto de combustível, verificar sua distância em relação ao entregador e por fim fornecer ao usuário a distância (dentro de um raio calculado em quilômetros) de um ou mais postos que estão próximos à distância pesquisados.

5 DESENVOLVIMENTO DA APLICAÇÃO

A aplicação que foi desenvolvida neste trabalho utilizou a tecnologia Java (J2ME) com a configuração CLDC e o perfil MIDP. A aplicação roda em um dispositivo móvel (celular) e está dividida em duas partes: uma aplicação do cliente (usuário) e uma aplicação do software servidor. A primeira é executada em um emulador de celular, e a segunda em um servidor web.

O software cliente permite ao usuário efetuar consultas via *Web* a uma base de dados georeferenciada, funcionando como um localizador geográfico de objetos. As consultas realizadas pelo software cliente são enviadas via web ao software servidor, onde um *Servlet* recebe os parâmetros da localização e por meio de funções existentes no POSTGIS identifica qual o estabelecimento está mais próximo do local onde o usuário está (informação fornecida pelo usuário ao entrar no sistema). Depois, o *Servlet* envia a resposta ao software cliente, onde os resultados (distância, nome de postos de gasolina e etc.) são mostrados ao usuário.

Devido a complexidade da utilização da API LBS, esta não foi implementada no trabalho proposto. Eram necessárias muitas outras classes para controle da aplicação e realização de cálculos para a localização exata de um local, além disso, deveria estar integrada com a utilização de mapas geográficos, para o apontamento do local solicitado. A sua utilização foi deixada para uma implementação futura.

A aplicação cliente (usuário) é executada em um emulador, na qual simula um dispositivo móvel com suas funcionalidades e particularidades.

5.1 Requisitos principais da aplicação

A aplicação desenvolvida neste trabalho teve como principais objetivos:

- estudar conceitos acerca de GPS e banco de dados geográficos;

- apresentar e utilizar os recursos de conectividade existentes na API J2ME;

- fazer com que o software efetue consultas em um servidor WEB passando como parâmetro, os dados colhidos através do software cliente;

- Tratar as consultas recebidas no servidor WEB, retornando apenas as informações condizentes com a localização informada;

- Apresentar a comunicação com serviços residentes em servidores, para retornar as consultas solicitadas;

- Apresentar consultas a dados geoprocessados, em um servidor web, para obter e determinar estabelecimentos mais próximos da posição em que se localiza o objeto solicitante.

5.2 Fluxograma para aplicação

Para a especificação da aplicação adotou-se o fluxograma da figura 5.1, a fim de demonstrar o fluxo dos principais processos executados nas aplicações.

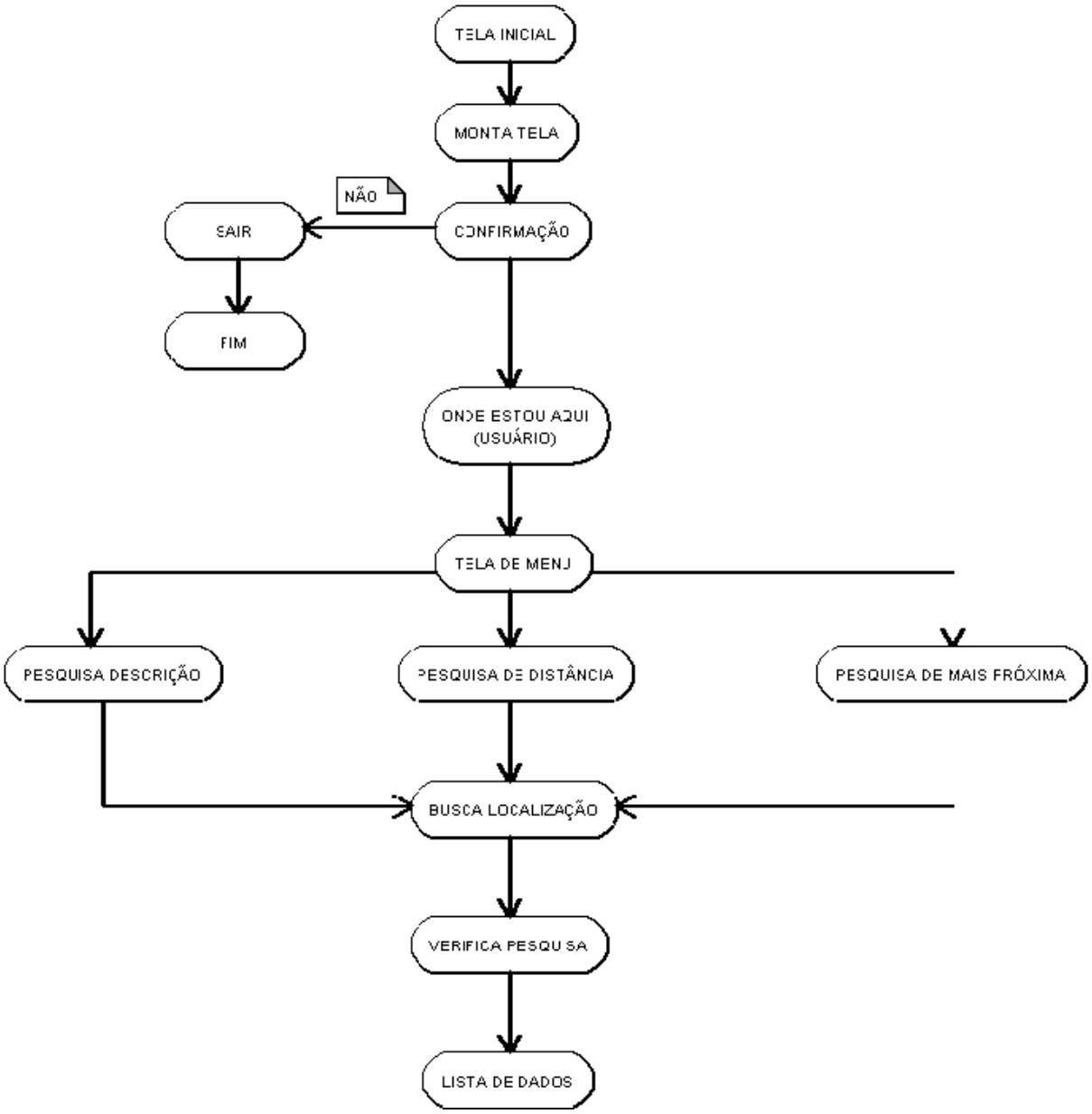

FIGURA 5.1 – Fluxograma da aplicação

5.3 Arquitetura da aplicação

A figura 5.2 ilustra a arquitetura da aplicação desenvolvida, mostrando ser possível manipular e recuperar informações do servidor de banco de dados, acessando um servidor web (na internet ou local) através de seu endereço (url), por meio de um dispositivo móvel.

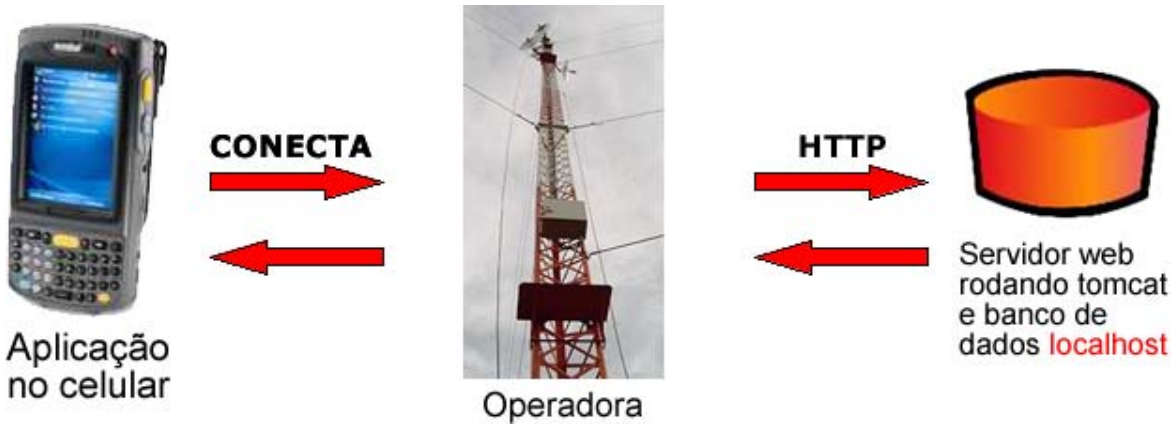

FIGURA 5.2 - Arquitetura da aplicação

Nesta aplicação específica utilizou-se como servidor de banco de dados o software PostgreSQL, junto com sua extensão geográfica PostGIS, conforme Figura 5.3.

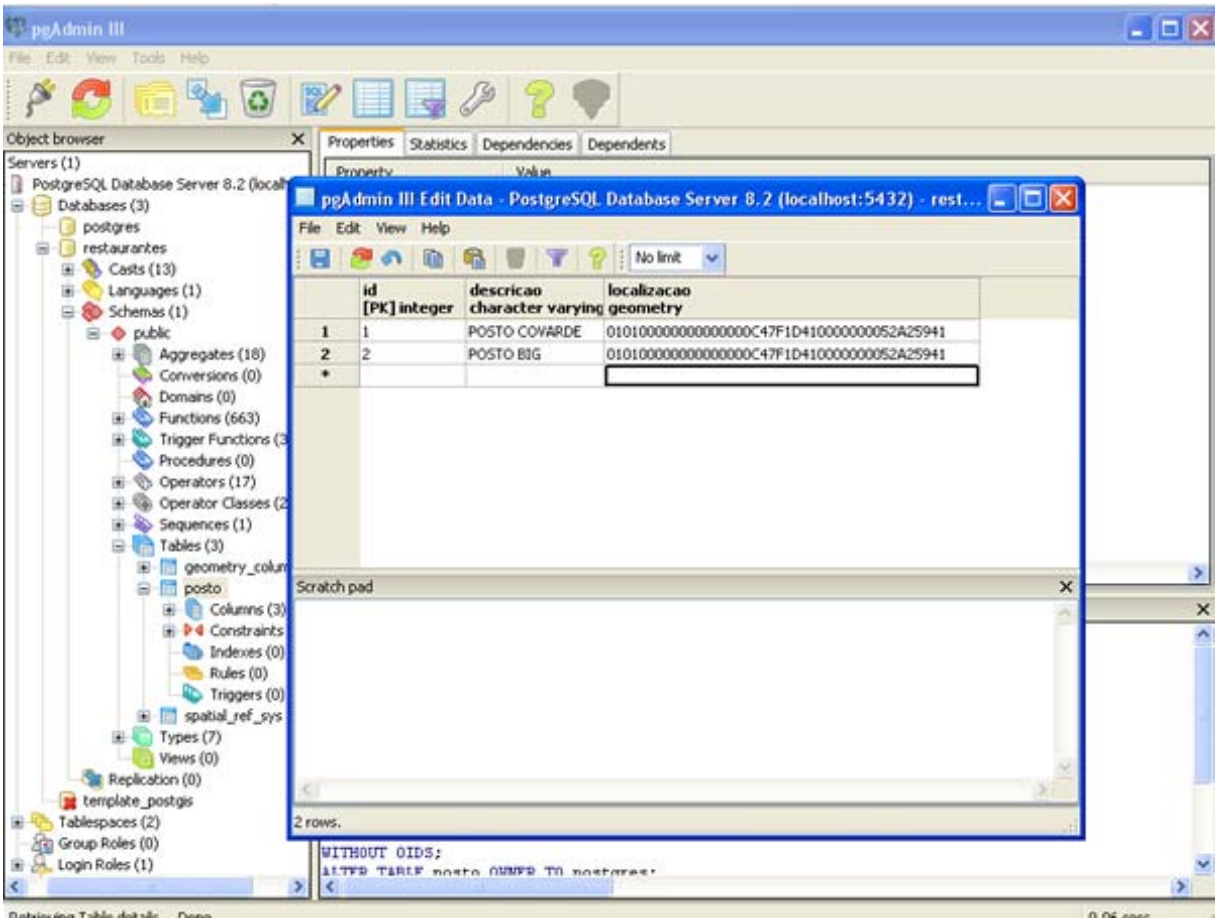

FIGURA 5.3 – Acessando dados georeferencia armazena no PostgreSQL (PostGIS)

A Figura 5.4 exibe um trecho da aplicação, na qual pode-se identificar a classe acessoWeb, responsável por implementar a conexão com o banco de dados no servidor web.

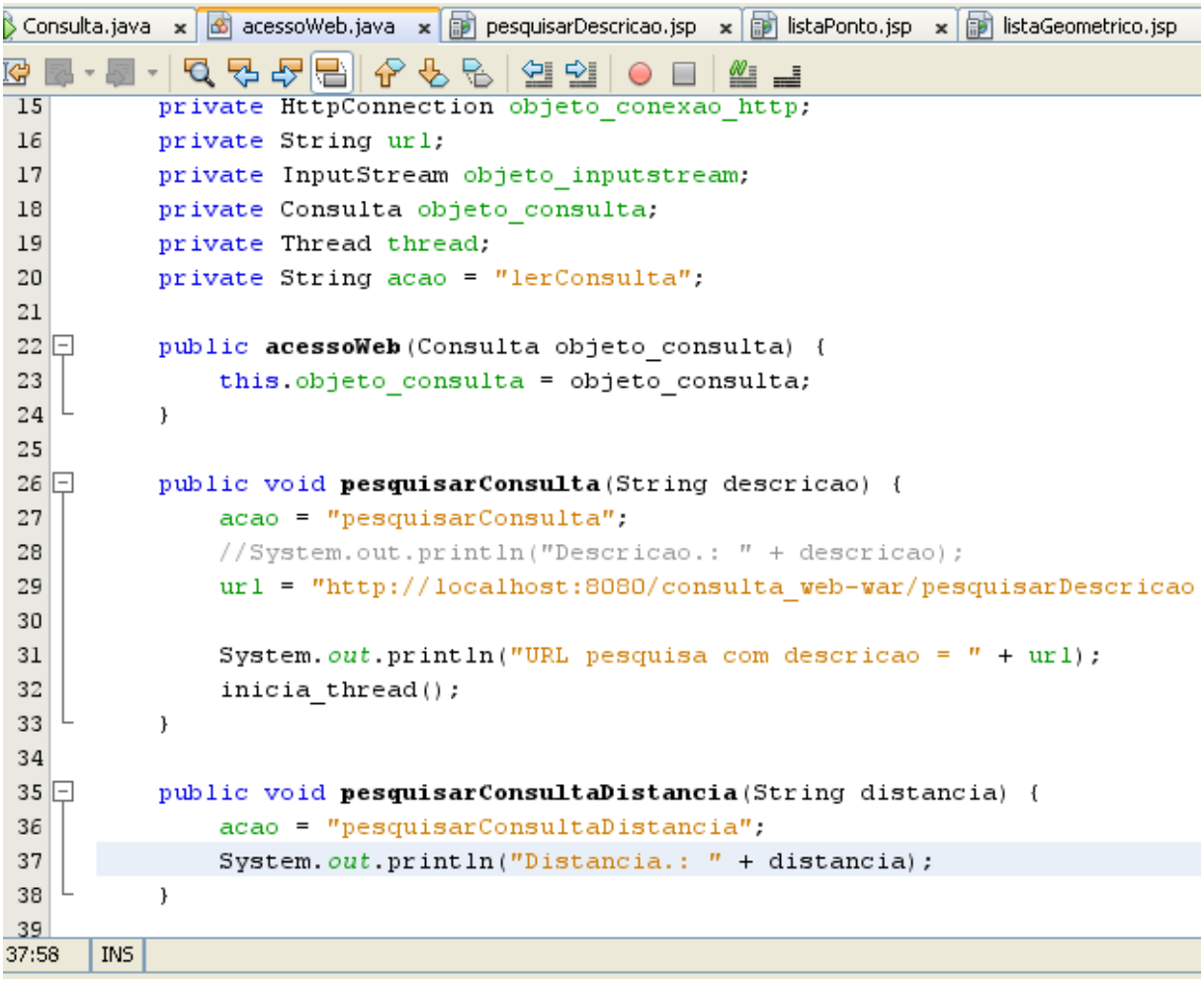

FIGURA 5.4 – Classe que implementa a conexão o banco de dados

Após conectar-se ao banco de dados o usuário pode manipular a aplicação no celular a fim de fazer suas consultas, através do celular. A figura 5.5 mostra uma consulta sendo executada na aplicação após a conexão com o banco de dados.

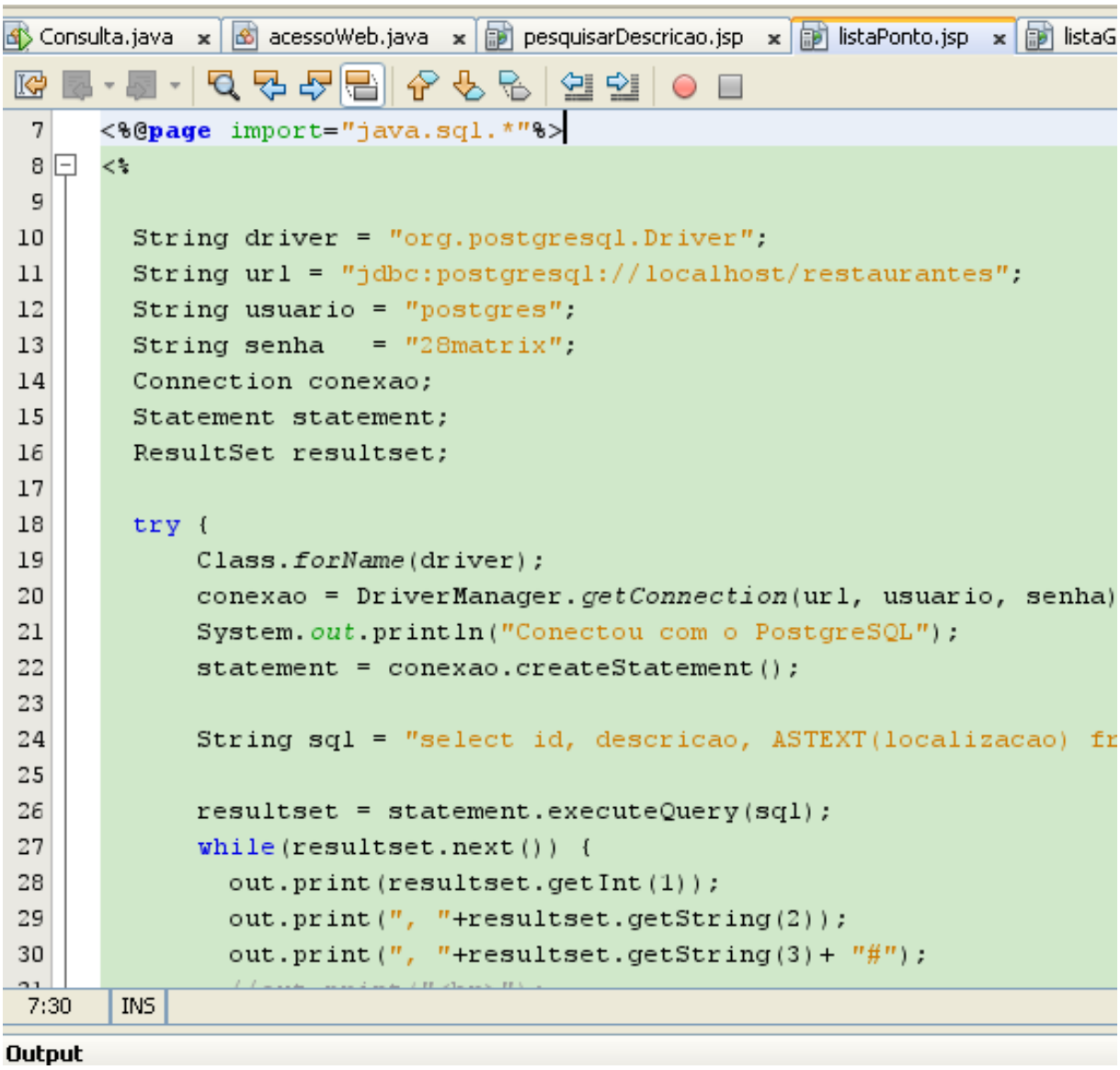

FIGURA 5.5 – Consulta sendo implementada no banco de dados

5.4 Implementação

Para ilustrar uma aplicação com a arquitetura proposta na seção anterior, resolveu-se criar uma tabela no POSTGIS com a seguinte estrutura:

Tabela 5.1: Tabela de Posto

Nome atributo	*Tipo atributo*
ID	INTEGER NOT NULL PRIMARY KEY
DESCRICAO	VARCHAR(40) NOT NULL
LOCALIZACAO	GEOMETRY

Desta forma foi possível:

- Localizar o posto de gasolina mais próximo em relação ao local informado pelo usuário ao iniciar o sistema. Para tanto, foram cadastradas as latitudes e longitudes de vários postos de gasolina localizados na região.

- Como pode ser visualizado na figura 5.8, estes postos de gasolina estão cadastrados em uma base de dados com suas descrições. No servidor web, existe um *Servlet* chamado Consulta, que fica aguardando as requisições vindas do dispositivo móvel. Nota-se pela figura 5.8 que um único objeto (POINT) armazena as informações da posição (latitude e longitude) e o outro atributo guarda o nome dos postos de gasolina.

- A tabela Posto é preenchida através de uma interface, onde o usuário informa latitudes e longitudes de diversos pontos ou ainda, pode ter seus dados guardados através do uso do GPS. Nesta última forma, conecta-se à porta COM1, o receptor GPS.

- O cliente (usuário) consiste em um *MIDlet* chamado ConsultaMIDlet. Ele está presente no dispositivo móvel, e é responsável por coletar as informações da latitude e longitude do dispositivo, que são gravadas respectivamente nos atributos de classe lat e lon após a execução do método pesquisarConsultaPonto. Este método é responsável por determinar no conjunto de caracteres recebidos do GPS, qual o valor da latitude a ser gravado no atributo lat e qual o valor da longitude a ser gravado no atributo lon. Com as informações de localização, o cliente (usuário) faz uma pesquisa ao servidor passando a latitude e longitude como parâmetros. O retorno desta pesquisa mostra qual a posto de gasolina mais próximo da localização informada.

5.4.1 Ferramentas utilizadas

Para programação do sistema foi utilizada a linguagem de programação Java, versão 6.0. Para o desenvolvimento do sistema *Web* (*Servlet*) e do *Midlet* para o dispositivo móvel, foi utilizada a ferramenta *IDE NetBeans*, versão 6.5, com o pacote de desenvolvimento móvel *Mobility Pack* (NETBEANS, 2008). Este pacote possui todas as ferramentas necessárias para o desenvolvimento de um aplicativo móvel. O emulador de dispositivo móvel integrado é representado na figura 5.6. Ele simula uma aplicação rodando em um ambiente real, tornando fácil o desenvolvimento do aplicativo.

Para o banco de dados, foi utilizada uma extensão do *PostgreSQL*, denominada *PostGIS*. Ele permite a manipulação de objetos GIS (Sistemas de Informação Geográfica) com o banco de dados.

5.4.2 Demonstrando o sistema

Após o sistema iniciar, a tela inicial é mostrada (figura 5.6), onde nela o usuário pode acessar o menu através do botão, (figura 5.7) localizado no canto superior direito do aparelho.

FIGURA 5.6 – Tela inicial

Uma mensagem de confirmação (figura 5.7) aparecerá para o usuário, questionando-o se realmente quer realizar a consulta. Se positivo, os dados então serão listados abaixo do campo de pesquisa. Se o usuário não optou por listar os dados, a página de pesquisa é novamente mostrada sem nenhuma alteração.

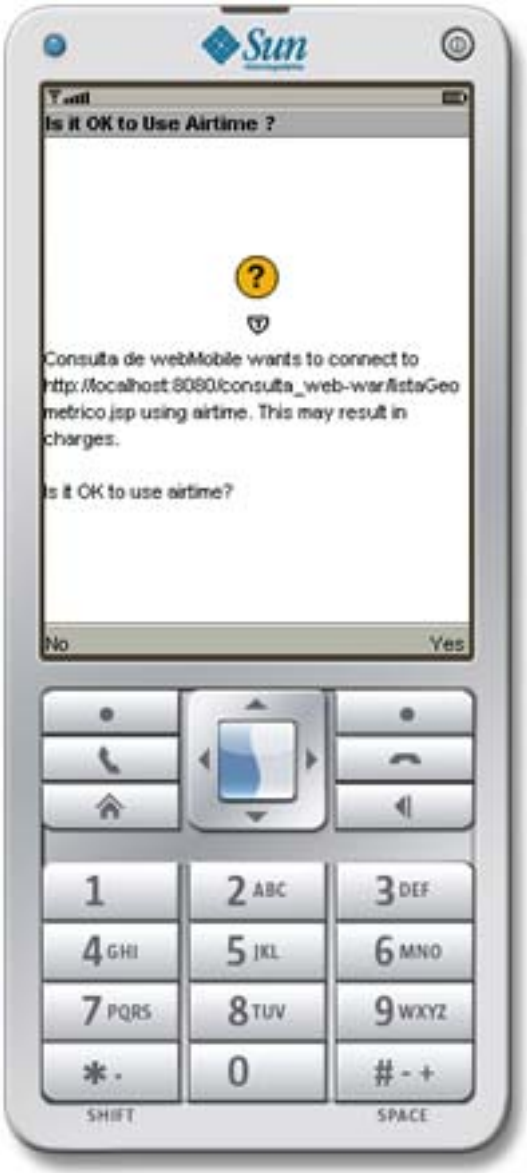

FIGURA 5.7 – Mensagem de confirmação

Após, o usuário pode selecionar o local onde ele se encontra, ou o local mais próximo dele, ilustrado na figura 5.8.

FIGURA 5.8 – Onde estou aqui

O menu (figura 5.9) exibirá as seguintes opções:

1 – Pesquisar por descrição

2 – Pesquisar distância

3 – Pesquisar mais próximo

FIGURA 5.9 - Tela de Menu

Se o usuário selecionar o primeiro item e acessá-lo através da tecla central do dispositivo (figura 5.9), o sistema abrirá a tela. Nesta, há um campo “descrição” que deve ser preenchido com a descrição do objeto que o usuário deseja encontrar. Após a digitação da informação é necessário que o usuário confirme, pressionando a tecla de pesquisa (figura 5.10), para que os dados sejam listados conforme o filtro informado.

FIGURA 5.10 - Lista de descrição

A segunda opção é de “Pesquisar distância”. Se acessá-la através do botão principal do aparelho, abrirá a tela de pesquisa (figura 5.11) com um campo para preenchimento da distância a ser pesquisada. Depois de informada e efetuada a pesquisa através do botão pesquisar, o sistema deverá encontrar todos os pontos existentes no raio da distância informada. Após realizada a busca, os dados serão listados abaixo do campo de pesquisa.

No exemplo abaixo, está sendo mostrada tela de pesquisa por distância. O usuário está informando a distância no campo do filtro (11 km) e a seguir pressionou o botão de pesquisa para que o sistema busque o posto de combustível mais próximo do local onde está (informação obtida na estrada do sistema). Neste caso, o usuário

informou uma distância de 11 km e o sistema buscou todos os postos dentro deste raio. A informação obtida foi o Posto de Assistência Feminina Bom Pastor, conforme ilustrada na figura abaixo.

FIGURA 5.11 - Lista de distância

A terceira opção “Pesquisar mais próximo” é acessada pelo menu principal. Após o acesso, a tela de listagem é carregada, mostrando a longitude e latitude do usuário e mostra a localização do ponto mais próximo cadastrado (figura 5.12), com a descrição, latitude e longitude. O usuário tem a opção de voltar a tela principal pressionando o botão voltar.

FIGURA 5.12 - Lista de mais próxima

5.5 O banco de dados

O OpenGIS (*Simple Features Specification for SQL*) define tipos de objetos padrão GIS, as funções requeridas para manipulá-las e uma especificação de tabela de meta-dados. Para assegurar que os meta-dados permaneçam consistentes, operações como criar e remover uma coluna espacial são carregadas através da definição de funções especiais definidas por *OpenGIS*.

Existem duas tabelas de meta-dados *OpenGIS*: SPATIAL_REF_SYS e GEOMETRY_COLUMNS. A tabela SPATIAL_REF_SYS carrega os identificadores

numéricos e descrições textuais de sistemas de coordenadas usados no banco de dados espacial.

O PostGIS conta com um grande número de funções para análises espaciais/topológicas que estendem o próprio SQL do PostgreSQL.

Ao utilizar um BDG (Banco de Dados Geográficos), um dos objetivos que se tem é ter a capacidade de realizar análises espaciais através do próprio banco. Por exemplo, espera-se que seja possível determinar intersecções entre elementos espaciais de camadas diferentes, armazenadas no BDG. Um exemplo prático seria efetuar uma consulta no banco buscando por todos os postos de gasolina. A resposta será o cruzamento entre as informações da área da cidade e dos pontos onde existem postos de gasolina.

Para este trabalho, foram selecionadas algumas funções, tendo em vista as demandas cotidianas das instituições que tratam da Geoinformação, bem como, a complexidade computacional das mesmas:

- *Contains*: retorna 1 (verdadeiro) se a geometria A "*Spatially Contains*" (contém espacialmente) a geometria B.

- *Distance*: retorna a distância cartesiana entre duas geometrias em unidades projetadas. Em outras palavras, é a circunferência que abrange todos os sentidos geográficos a a partir de uma base, que neste caso é localização do usuário.

- *Buffer*: Retorna um valor geométrico que representa todos os pontos cuja distância a um outro valor geométrico é menor ou igual a uma determinada distância.

O banco de dados é composto por quatro tabelas (GEOMETRY_COLUMNS, SPATIAL_REF_SYS, POSTO e USUARIO). A tabela GEOMETRY_COLUMNS armazena os dados referentes a localização geométrica dos objetos. A SPATIAL_REF_SYS armazena os dados de cálculos realizados pelo sistema (cálculos geométricos). A tabela POSTO armazena os dados dos postos de combustíveis cadastrados no sistema e a tabela USUÁRIO armazena os dados dos usuários que cadastraram os pontos de localização (posto de combustível). A estrutura das tabelas pode ser visualizada no dicionário de dados abaixo:

Tabela 5.2: Tabela de SPATIAL_REFS_SYS

Esta tabela armazena os tipos Sistema de Referência Espacial utilizados.

Nome atributo	*Tipo atributo*
SRID	INTEGER NOT NULL PRIMARY KEY
AUTH_NAME	VARCHAR (256)
AUTH_SRID	INTEGER
SRTEXT	VARCHAR (2048)

Tabela 5.3: Tabela de GEOMETRY COLUMNS

Esta tabela armazena os dados de coordenadas da Feição.

Nome atributo	*Tipo atributo*
Feature_TABLE_CATALOG	VARCHAR (256) NOT NULL
Feature_TABLE_SCHEMA	VARCHAR (256) NOT NULL
Feature_TABLE_NAME	VARCHAR (256) NOT NULL
Feature_GEOMETRY_COLUMN	VARCHAR (256) NOT NULL
Geometry_TABLE_CATALOG	VARCHAR (256) NOT NULL
Geometry_TABLE_SCHEMA	VARCHAR (256) NOT NULL
Geometry_TABLE_NAME	VARCHAR (256) NOT NULL
STORAGE_TYPE	VARCHAR (256) NOT NULL
GEOMETRY_TYPE	INTEGER
COORD_DIMENSION	INTEGER NOT NULL
MAX_PPR	INTEGER
SRID	INTEGER NOT NULL

Tabela 5.4: Tabela de Posto

Esta uma Geometria que está sendo referenciada na tabela de Posto

Nome atributo	*Tipo atributo*
ID	INTEGER NOT NULL PRIMARY KEY
DESCRICAO	VARCHAR (40) NOT NULL
LOCALIZACAO	GEOMETRY

Para a primeira consulta (Pesquisar por descrição), é passado como parâmetro na tela de pesquisa o nome do objeto, e espera-se como retorno o código e descrição do objeto, em ordem alfabética, a seguinte consulta realizada no banco:

```
"SELECT ID, DESCRICAO, ASTEXT(LOCALIZACAO) " +
"FROM POSTO " +
"WHERE DESCRICAO LIKE '%"+ request.getParameter("descricao") +"%' " +
"ORDER BY DESCRICAO ";
```

A segunda consulta (Pesquisar distância) é realizada com o objetivo de buscar no banco, os objetos cujo ponto esteja dentro do raio (distância em quilômetros) informado pelo usuário. O usuário informa na tela de pesquisa apenas o valor da distância e o sistema verifica quais pontos estão dentro do raio informado.

```
"SELECT ID, DESCRICAO, ASTEXT(LOCALIZACAO) " +
"FROM POSTO " +
"WHERE CONTAINS(buffer(GeomFromText('"+request.getParameter("ponto")+"'),
"+request.getParameter("distancia")+"), LOCALIZACAO)";
```

A terceira consulta (Pesquisar mais próximo) é realizada com o objetivo de buscar o ponto cadastrado pelo usuário que esta realizando a consulta, mais perto de onde ele está. A consulta busca pela localização do usuário e retorna os dados (código e descrição) do ponto mais próximo e do ponto de onde partiu a consulta.

```
"SELECT P1.ID, P1.DESCRICAO, P2.DESCRICAO, " +
"DISTANCE(P1.LOCALIZACAO, P2.LOCALIZACAO)/1000 " +
"FROM POSTO P1, POSTO P2 " +
"WHERE P1.ID > P2.ID " +
"AND ASTEXT(P1.LOCALIZACAO) = '"+request.getParameter("ponto")+"' ";
```

5.6 Dificuldades encontradas

A API LBS foi de difícil entendimento para mim, por isso deixarei o seu uso para uma implementação futura nesta aplicação. A tecnologia necessita de muitas classes de controle e execução dos seus métodos básicos de localização. O sistema também é dependente de várias bibliotecas que não fazem parte do ambiente padrão de desenvolvimento de aplicativos móveis, por exemplo a integração com mapas geográficos para a localização de um ponto com apontamento do local exato no mapa.

Outro ponto importante é que foi necessário adquirir um aparelho compatível com a tecnologia Java e sistema GPS para testes da aplicação em ambiente real.

6 CONCLUSÃO

Neste trabalho foi abordada uma API para serviços baseados em localização. Por meio desta API, as aplicações desenvolvidas, possibilitam obter, transferir, armazenar e gerenciar informações de localização de dispositivos móveis, extraindo informações geográficas através de sistemas GIS e fornecendo serviços baseados em localização. Inicialmente, realizou-se uma pesquisa sobre as tecnologias de localização disponíveis e mais utilizadas, como: métodos de localização, APIs de localização, de mobilidade e sistemas GIS.

Nos estudos foi observado que desenvolver aplicações GPS para dispositivos sem fio representam muitos desafios aos programadores, projetistas, etc, pois as arquiteturas destes dispositivos são bem mais limitadas que as de um PC. Além desta limitação, existe o fato de que a interação com a internet por meio de um dispositivo móvel é muito diferente de um computador pessoal, pois as atividades realizadas com um celular são fundamentalmente diferentes das realizadas em frente de um PC, pois a capacidade de processamento é muito mais limitado em um dispositivo móvel, tendo em vista sua baixa capacidade de memória e velocidade de processamento.

As APIs de localização são disponibilizadas nos dispositivos pelos seus fabricantes (Nokia, LG, Siemens e outros), mas podem ser criadas por terceiros (Sun, IBM, Oracle, etc.). Com a pesquisa inicial viu-se que uma API genérica de terceiro, baseada na plataforma J2ME, a *API Location*, está em grande desenvolvimento e representa amplas oportunidades para diferentes provedores de localização. No entanto, existem no mercado outras APIs alternativas e/ou complementares à *API Location* de J2ME. Além disso, existe outra possibilidade de que a plataforma J2ME não esteja disponível para determinados dispositivos. Por isso, parece ser necessário modelar uma API genérica que disponibilizasse compatibilidade de programação com todas as APIs disponíveis no mercado. Foi encontrada uma API chamada LBS, que centralizou as

funções de tratamento das informações de localização necessárias, utilizadas por serviços baseados em localização.

As aplicações desenvolvidas neste trabalho conseguir os objetivos. Nesses foi possível estabelecer a comunicação entre o software do emulador e o banco de dados, baseando-se numa conexão e uma comunicação com um Servlet, baseando-se numa conexão HTTP. O uso de estas duas conexões tornarem possível um bom estudo sobre o framework de conectividade do J2ME. Através da comunicação do software do emulador, para buscar a localização, e posterior comunicação com o Servlet a aplicação pode cumprir seus requisitos e mostrar o conceito de uma aplicação de serviços baseados em localização.

Finalmente, após definida a arquitetura da solução, objetiva-se iniciar o desenvolvimento do protótipo utilizando a API apresentada no trabalho, tendo como objeto de estudo, uma aplicação em um telefone celular, no qual o usuário poderá fazer consultas via WEB a uma base de dados georeferenciada, interagindo de modo transparente com conceitos acerca de localização geográfica e banco de dados geográficos.

7 REFERÊNCIAS BIBLIOGRÁFICAS

BAÇÃO, Fernando Lucas; COSTA, José Jesus. **O Papel do Data Mining Geo-espacial nos Location Based Services.** Portugal, p.4-5. Disponível em: http://www.isegi.unl.pt/docentes/fbacao/Costa_CAPSI03.pdf. Acesso em: 14 de março de 2008.

CÂMARA, G. et al. **Anatomia de sistemas de informação geográfica.** Rio de Janeiro: SBC,
1996. Disponível em : <http://www.dpi.inpe.br/gilberto/livro/anatomia.pdf>. Acesso em: 03 de maio de 2008.

CAMARA, G. **Análise de Arquiteturas para Banco de Dados Geográficos Orientados a Objetos.** São José dos Campos, Instituto Nacional de Pesquisas Espaciais (INPE), Tese de Doutorado. 1994.

CÂMARA, G. et al. **Banco de Dados Geográficos.** São Paulo: MundoGEO, 2005. 506p.

CÂMARA, G. **Modelos, Linguagens e Arquiteturas para Bancos de Dados Geográficos.** Ph.D., Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, SP. 1995.

CARDOSO, J. **Java para Telemóveis MIDP 2.0.** 2006. Disponível em: <livromidp.jorgecardoso.org/>. Acesso em: 23 de maio de 2008.

CASANOVA, A. Marco. **Bancos de Dados Geográficos.** Curitiba. MundoGEO, 2005.

DEITEL, H. M. **Internet & world wide web, como programar.** Porto Alegre: Bookman, 2003.

DEITEL, H. M. **Java, como programar.** Porto Alegre: Bookman, 2001.

DEPINÉ, Fábio Marcelo. **Protótipo de software para dispositivos móveis utilizando Java ME para cálculo de regularidade em rally.** 2002. 55 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

ELMASRI, R; Navathe, S.B. **Sistemas de Banco de Dados.** São Paulo: Pearson Education do Brasil, 2005.

FALEIROS, Marco Antônio C.. **Localização sem fio - Serviços baseados em localização com J2ME e a Location API.** Disponível em: <http://www.devmedia.com.br/visualizacomponente.aspx?COMP=431&SITE=5>. Acesso em: 2 de junho de 2008.

FERREIRA, Karine Reis. **Interface para Operações Espaciais em Banco de Dados Geográficos.** Mestrado, Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, SP. 2003.

GARDELS, K., 1996. **The Open GIS Approach to Distributed Geodata and Geoprocessing.** In: Third International Conference/Workshop on Integrating GIS and Environmental Modeling. Santa Fe, NM, USA, 1996. p. 21-25.

GEOGRAPHIC information system. Disponível em: <http://en.wikipedia.org/wiki/Geographic_Information_System>. Acesso em: 12 de maio
2008.
GIS, Disponível em: <http://www.gis.com/whatisgis/index.html>. Acesso em: 12 de maio de 2008.

HAIGES, Sven. **The Location API.** Disponível em: <http://jdj.syscon.com/read/37747.htm>. Acesso em: 17 de maio de 2008.

HEMPHILL, D. A.; WHITE, J. P. **Java 2 Micro Edition. [S.l.]: Manning Publications Company**; ISBN 1930110332, 2002. 400 p.

JAVADOC Package javax.microedition.location. Disponível em: <http://wwwusers. cs.umn.edu/~czhou/docs/jsr179/lapi/javax/microedition/location/packagesummary. html>, <http://www.forum.nokia.com/java/jsr179#jsr179>. Acesso em: 14 de abril de 2008.

JÚNIOR, Edvaldo Simões da Fonseca. **Global positioning system.** São Paulo, [2003]. Disponível em: <http://www.ptr.usp.br/Cursos/SIG_GPS/gps/home.htm>. Acesso em 4 de maio de 2008.

KLIEMANN, José M.. **Monografia Modelagem de uma API para serviços baseados em localização inegrada com APIs de localização, o middleware EXEHDA e GIS.** UFRGS, 2006.

JOHNSON, Thienne M. **Java para dispositivos móveis: desenvolvendo aplicações com J2ME.** Novatec, 2007.

LOUREIRO, A.A.F.; MATEUS, G.R.; PÁDUA, C.I.P.S. et al; **Serviços em Ambientes Móveis.**

LOYTANA, Kimmo. **JSR 179: Location API for J2ME. Java Community Process.** Disponível em: <http://jcp.org/en/jsr/detail?id=179>. Acesso em: 21 de maio de 2008.

MAHMOUD, Qusay H. **J2ME and Location-Based Services.** Disponível em:

<http://developers.sun.com/techtopics/mobility/apis/articles/location/index.html>. Acesso em: 19 de maio de 2008.

MENEZES, Arionaldo R.; MEDEIROS, Sandro L. **GIS na PETROBRAS.** Revista INFOGEO. Curitiba, v.7, n37, set 2005.

MICROSYSTEMS, S. **Connected Limited Device Configuration Specification.** version 1.1. [S.l.], 2003.

MICROSYSTEMS, S. **Connected Device Configuration Specification.** version 1.1.2. [S.l.],
2006.

MICROSYSTEMS, S. **Java Platform, Micro Edition (Java ME).** 2006. Disponível em:
<http://java.sun.com/javame/>. Acesso em: 23 de maio de 2008.

MIRANDA, R. B.; MARCONDES, V. P. P. **Prime: uma solução Java para acesso móvel**
a informações utilizando GSM/GPRS. Santa Rita do Sapucaí, MG, 2004.

MUCHOW, John W. **Core J2ME – Tecnologia e MIDP.** São Paulo, SP. Markon Books, 2004.

NOKIA-a. **MIDP: Location API Developer's Guide.** , 2006. Disponível em:
<http://sw.nokia.com/id/708ac992-1168-43b2-a46aaa3931e49d48/
MIDP_Location_API_Developers_Guide_v1_0_en.pdf>. Acesso em: 25 de maio de 2008.

OGC, 2005. **OGC to Begin Geospatial Semantic Web Interoperability Experiment.** http://www.opengeospatial.org/press/?page=pressrelease&year=0&prid=222. Acesso em: 23 de maio de 2008.

ORACLE. **Java programming.** World Headquarters: Oracle Corporation, 2002.

QUALCOMM. Snap Track. 2003. Disponível em:
<http://www.snaptrack.com/index.jsp>.
Acesso em: 04 de maio de 2008.

PERCIVALL, G. (ed), 2003. OpenGIS® Reference Model. Document number OGC 03-040 Version: 0.1.3. Open Geospatial Consortium, Inc.

PINHEIRO, C. **J2ME - Java para os portáteis.** 2003. Disponível em:
<http://www.imasters.com.br/artigo/1539>. Acesso em: 18 de maio de 2008.

POSTGIS. **PostGIS Manual.** Disponível em:
http://postgis.refractions.net/docs/postgis.pdf
Acesso: 18 de maio de 2008.

REGGIANI, Lúcia. **Admirável mundo sem fio.** Info Exame, São Paulo, v.1, n.212,p.23, set.
2003.

ROCHA, Mauro Nacif. **Serviços baseados em localização.** Viçosa, [2001]. Disponível em:
<http://www.dpi.ufv.br/~nacif/cmovel/Aula16.pdf >. Acesso em: 17 de abril de 2008.

SANTOS, André Milke dos. **Artigo Extensões Espaciais de Sistemas Gerenciadores de Banco de Dados.** 2006/01.

SILBERSCHATZ, Abraham et al. **Sistema de Banco de Dados.** São Paulo: Pearson Education do Brasil, 1999. 778p.

SHARMA, C.; **Aplicações Comerciais da Internet sem Fio: wireless technology.** São Paulo: Makron Books, 2001. 285 p.

SILVA, Miguel Reis Castilho da. **Uma introdução sobre sistemas de localização.** Évora,
[2003]. Disponível em: <http://alunos.uevora.pt/~l12058/es/t3/>. Acesso em: 04 de abril de 2008.

SOUJAVA. **Sociedade de usuários Java.** [S.l.], [2003]. Disponível em:
<http://www.soujava.org.br/>. Acesso em: 16 de maio de 2008.

TAFFE, F.A.; **Análise de Requisitos e ferramentas para implementação de sites de comercio eletrônico móvel.** Florianópolis: Departamento de Ciência da Computação – UFSC, 2002.

ZURSTRASSEN, Leonardo. **Location based services.** [S.l],[2003]. Disponível em:
<http://www.wirelessbrasil.org/wirelessbr/colaboradores/zurstrassen/lbs_01.html>. Acesso
em: 28 de março de 2008.