UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA – CCET
CURSO DE PÓS-GRADUAÇÃO EM ENGENHARIA DE ELETRICIDADE

ERICH FARIAS MONTEIRO

UM FRAMEWORK PARA O GERENCIAMENTO DA INFORMAÇÃO DE LOCALIZAÇÃO

São Luís
2005

ERICH FARIAS MONTEIRO

UM FRAMEWORK PARA O GERENCIAMENTO DA INFORMAÇÃO DE LOCALIZAÇÃO

Dissertação apresentada ao curso de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica na área de Ciência da Computação.

Orientadores: Prof. Dr. Anselmo Cardoso de Paiva
Prof. Dr. Francisco José da Silva e Silva

São Luís
2005

Monteiro, Erich Farias

Um Framework para o Gerenciamento da Informação de Localização / Erich Farias Monteiro. – São Luís, 2005.

110 f. : il.

Dissertação (Mestrado em Engenharia de Eletricidade – Área de Ciência da Computação) – Universidade Federal do Maranhão, 2005.

1. Sistemas baseados em localização – Rastreamento. 2. Framework. 3. Objetos móveis. 4. Redes sem fio. I. Título

CDU 621.397.3

ERICH FARIAS MONTEIRO

UM FRAMEWORK PARA O GERENCIAMENTO DA INFORMAÇÃO DE LOCALIZAÇÃO

Dissertação apresentada ao curso de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica na área de Ciência da Computação.

Aprovada em: ___ / ___ / _____

BANCA EXAMINADORA

_____

Prof. Dr. Anselmo Cardoso de Paiva
Universidade Federal do Maranhão
(Orientador)

_____

Prof. Dr. Francisco José da Silva e Silva
Universidade Federal do Maranhão
(Orientador)

_____

Prof. Dr. Cláudio de Souza Baptista
Universidade Federal de Campina Grande

_____

Prof. Dr. Aristófanes Corrêa Silva
Universidade Federal do Maranhão

Aos meus pais, Luz de Maria e José Emilio.

Aos meus irmãos Eloy e Elano.

Ao meu grande amor Camila Melo.

AGRADECIMENTOS

Aos meus orientadores, Anselmo Paiva e Francisco Silva, por toda paciência e dedicação despendidas para realização deste trabalho.

Aos meus queridos pais, Emílio e Luz de Maria, pelo esforço realizado para me proporcionar toda a infra-estrutura necessária para a realização desta conquista.

A toda minha família, em especial aos meus irmãos Elano e Eloy, pela amizade e pelo estímulo dado ao longo da minha vida e deste trabalho.

À Camila, pelo companheirismo e apoio recebidos ao longo desses anos.

Aos meus amigos, por compartilharem comigo a felicidade de mais uma conquista.

Ao David Cavassana, pela ajuda na implementação do estudo de caso deste trabalho.

Ao Alcides, pelo apoio administrativo prestado durante todo o curso de mestrado.

A todos os professores do curso de Mestrado e de Ciência da Computação, que não mediram esforços para transmitir aos seus alunos o verdadeiro conhecimento.

Enfim, agradeço a todos que de uma alguma forma contribuíram para a realização deste trabalho e me ajudaram no decorrer do curso.

RESUMO

Apresenta-se o estado atual das tecnologias para a localização de objetos móveis e para o desenvolvimento de sistemas baseados em localização (LBS). Realiza-se uma revisão das arquiteturas e dos requisitos funcionais para a construção de aplicações baseadas em localização e propõe-se uma arquitetura para essas aplicações onde a gerência da informação de localização está desacoplada e pode ser facilmente reutilizada, levando à proposta e implementação de um *framework* reutilizável para a gerência da informação de localização, que disponibiliza funcionalidades para a gestão eficiente da localização dos objetos móveis. Finalmente, avalia-se a utilização do *framework* proposto através do desenvolvimento de uma aplicação LBS para o rastreamento dos usuários que estiverem transitando na região do centro histórico da cidade de São Luís, disponibilizando recursos de rastreamento, determinação de proximidade e consultas de rotas percorridas pelo usuário.

Palavras-chave: Sistemas Baseados em Localização, *framework*, objetos móveis, redes sem fio.

ABSTRACT

This work presents the actual stage for the mobile objects location and for the development of location based applications or services (LBS). We make a review of available system architectures and functional requirements to the development of location based applications. Also we suggest an architecture to this class of applications that encapsulate the location information management that is completely reusable. To implement this architecture we describe the implementation of a reusable framework to the location information management, that makes available functionalities to the efficient management of mobile objects location information, that is evaluated trough the development of an location based application to track mobiles objects in an historical site of São Luis city the capital of Maranhão.

Keywords: Location Based Service, *framework*, mobile objects, Wireless networks.

SUMÁRIO

LISTA DE FIGURAS **11**
LISTA DE SIGLAS **12**
1 INTRODUÇÃO **12**
2 SISTEMAS BASEADOS EM LOCALIZAÇÃO **16**
2.1 Arquitetura de Sistemas Baseados na Localização **18**
2.2 Requisitos de Aplicações Baseadas em Localização **22**
2.3 Tecnologias de localização de dispositivos móveis **23**
2.3.1 Soluções baseadas em rede *(network-based*) 24
2.3.1.1 Identificação de Célula (*Cell Identification - CID*) 24
2.3.1.2 Ângulo de Chegada (*Angle of Arrival* -AOA) 25
2.3.1.3 Tempo de Chegada (*Time of Arrival* -TOA) 27
2.3.1.4 Diferença do Tempo de Chegada (*Time Difference of Arrival* -TDOA) 28
2.3.1.5 Padrão de Trajetória (*Multipath Fingerprinting*) 29
2.3.2 Soluções Baseadas no Dispositivo (*Handset-Based*) 29
2.3.2.1 GPS (*Global Positioning System*) 30
2.3.2.2 AGPS (*Assisted-GPS*) 32
2.3.2.3 E-CID (*Enhanced Cell ID*) 33
2.3.3 Soluções Híbridas 34
2.3.4 Sistemas de posicionamento para ambientes fechados (*indoor*) 35
2.3.4.1 Localização Baseada em raios infravermelhos 35
2.3.4.2 Localização Baseada em Radiofreqüência 37
2.3.4.3 Localização Baseada em ultra-som 38
2.4 Aplicações de Sistemas Baseados na Localização **39**
2.4.1 E911 42
2.4.2 Serviços de Informação 43
2.4.3 Serviços de Rastreamento 44
2.4.4 Navegação 46
3 TRABALHOS CORRELATOS **48**
3.1 Arquiteturas **48**
3.2 Frameworks **54**
4 FRAGIL: FRAmework para o Gerenciamento da Informação de Localização **60**

4.1 Arquitetura do Framework.....60
4.1.1 *RegisterService* - Componente de registro do objeto móvel.....64
4.1.2 *UpdatePosService* - Componente de atualização da posição 65
4.1.3 *QueryService* - Componente de Consultas 68
4.1.4 *EventService* - Componente de Eventos.....71
4.2 Modelagem e Implementação do *Framework* 71
4.2.1 Casos de Uso.....72
4.2.2 Diagrama de Classes.....74
4.2.2.1 Pacote util 74
4.2.2.2 Pacote LS 76
4.2.2.3 Pacote ApplicationAPI 78
4.2.2.4 Pacote ClientAPI.....79
4.2.3 Diagrama de Seqüência 80
4.2.4 Modelagem de Dados 88
5 Estudo de Caso: Sistema de Rastreamento de Usuários Móveis (SRUM).....90
5.1 Descrição das Funcionalidades 90
5.2 Implementação 92
5.2.1 Módulo Cliente.....93
5.2.2 Ferramenta de Monitoramento.....95
5.2.3 Considerações Finais 100
6 CONCLUSÃO 102
REFERÊNCIAS 106

LISTA DE FIGURAS

Figura 2.1 – LBS como Convergência de várias Tecnologias (SHIODE *et al.*, 2002) 16
Figura 2.2 – Arquitetura Geral de um sistema LBS 18
Figura 2.3 – Técnica de Angulação 26
Figura 2.4 – Tempo de Chegada (NACIF, 2005) 27
Figura 2.5 – Rede satélite – GPS (SCHILLER, 2004)..... 30
Figura 2.6 – Enhance Observed Time Difference (E-OTD) 34
Figura 3.1 – Arquitetura Geral do OpenLS (OGC, 2004)..... 50
Figura 3.2 – Principais componentes da arquitetura..... 51
Figura 3.3 – Arquitetura do LSM (AGRE *et al*, 2002)..... 53
Figura 3.4 – Hierarquia de classes do *wireless framework*..... 55
Figura 3.5 – Arquitetura do *framework* abstrato (ANDERSEN *et al*, 2003)..... 56
Figura 3.6 – Arquitetura do UFL (LARA, 2003) 58
Figura 4.1 – Arquitetura geral do FRAGIL..... 61
Figura 4.2 – Diagrama de Casos de uso para o ator usuário móvel..... 72
Figura 4.3 – Diagrama de Casos de uso para o ator Aplicação LBSS..... 73
Figura 4.4 – Diagrama de Pacotes 74
Figura 4.5 – Diagrama de Classes do Pacote util 75
Figura 4.6 – Diagrama de Classes do Pacote LS 77
Figura 4.7 – Diagrama de Classes do Pacote AplicationAPI 79
Figura 4.8 – Classe ClientAPI 80
Figura 4.9 – Diagrama de Seqüência – Iniciar/Encerrar sessão e atualizar posição..... 81
Figura 4.10 – Diagrama de Seqüência – Realizar Consulta..... 84
Figura 4.11 – Diagrama de Seqüência – Registrar Evento 86
Figura 4.12 – Diagrama de Seqüência – Notificar Evento..... 87
Figura 4.13 – Diagrama de Dados 88
Figura 5.1 – Diagrama de Casos de Uso do SRUM 91
Figura 5.2 – Telas de interface da aplicação cliente 94
Figura 5.3 – Trecho de código exemplificando a realização da classe ClientAPI. 95
Figura 5.4 – Interface Gráfica da Ferramenta de Monitoramento. 97
Figura 5.5 – Ficha Rastrear Usuários..... 98
Figura 5.6 – Trecho do mapa mostrando os usuários móveis. 98
Figura 5.7 – Ficha Consultar Percursos. 99
Figura 5.8 – Representação de uma rota percorrida por um *mo.*..... 99
Figura 5.9 – Ficha Obter Proximidade.. 100

LISTA DE SIGLAS

AGPS	–	Assisted-GPS
ANI	–	Automatic Number Information
AOA	–	Angle of Arrival
AOI	–	Area of Interest
API	–	Application Programming Interface
CGI	–	Cell of Global Identity
CID	–	Cell Identification
CLDC	–	Connected Limited Device Configuration
E-CID	–	Enhanced Cell ID
E-OTD	–	Enhanced Observed Time Difference
EPC	–	Electronic Product Code
ERB	–	Estação Rádio-base
FCC	–	Federal Communications Commission's
FRAGIL	–	Framework para Gerenciamento de Informações de Localização
GMLC	–	Gateway Mobile Location Center
GMS	–	GeoMobility Server
GPS	–	Global Positioning System
LBS	–	Location-based Service or Systems
LBSS	–	LBS Server
LMU	–	Location Measurement Unit
LS	–	Location Server
LSM	–	Location Service Module
MIDP	–	Mobile Information Device Profile
MSC	–	Mobile Switching Center
NAVSTAR	–	Navigation System with Timing and Ranging
OGC	–	OpenGeoSpatial Consortium
PDA	–	Personal Digital Assistant
PDE	–	Position-Determining Equipment
PIM	–	Personal Information Management
POI	–	Point of Interest
PPS	–	Precision Positioning Service

RFID	–	Radio Frequency Identification
SA	–	Selective Availability
SIG	–	Sistemas de Informação Geográfica
SPS	–	Standard Positioning Service
TDOA	–	Time Difference of Arrival
TOA	–	Time Of Arrival
TRF	–	Transmissores de Rádio Freqüência
TTFF	–	Time to First Fix
ULF	–	Universal Location Framework
UML	–	Unified Modeling Language

1 INTRODUÇÃO

O advento da computação móvel trouxe consigo um conjunto de novas aplicações que se beneficiam da constante necessidade de acesso à informação, do menor custo da infra-estrutura de comunicação e da diminuição nos custos e miniaturização dos dispositivos móveis para conquistar um número cada vez maior de usuários.

A característica de mobilidade, intrínseca à computação móvel, abriu uma nova área para o desenvolvimento de aplicações. Associada à mobilidade, temos que a identificação da localização se torna naturalmente um atributo crítico, pois abre as portas para o desenvolvimento de uma grande variedade de novas aplicações e serviços. Os sistemas que se utilizam dessa informação são denominados sistemas baseados na localização (LBS). Mais precisamente, podemos definir LBS como aplicações que se utilizam da informação de localização para fornecer serviços contextualizados aos seus usuários (MALLICK, 2003).

A utilização da informação de localização do usuário permite a concepção de conceitos de serviços completamente inovadores, oferecendo informações ajustadas ao seu contexto (por exemplo**,** informações climáticas ajustadas à região onde o usuário se encontra), incrementando, assim, consideravelmente a utilidade dos serviços disponibilizados (SCHILLER;VOISARD, 2004). Entendemos que aplicações sensíveis à localização aumentam a eficácia dos serviços por fornecerem um acesso customizado aos dados baseados não somente nas preferências do usuário, mas também em sua posição atual. Isso eleva a personalização de conteúdo a um novo nível, fornecendo vários benefícios tanto para os usuários quanto para os desenvolvedores de aplicações.

Podemos verificar que, aliadas às características e benefícios anteriormente citados, a modernização das técnicas de localização e o crescente aumento no número de usuários de dispositivos móveis dão às aplicações baseadas em localização a tendência de um

crescimento exponencial, oferecendo informações cada vez mais precisas, objetivas e úteis. Pesquisa realizada por Shiode et al. (2002) demonstra a tendência de mercado e o potencial reservado para as aplicações centradas na localização do dispositivo, as quais, a cada ano, ganham fundamental importância entre os usuários, revelando-se uma área de domínio entre as aplicações de dispositivos móveis. Em resumo, pode-se afirmar que, em termos de negócio de conteúdos, a exploração da informação posicional tem o potencial de encaixar o contexto geográfico do usuário como uma das variáveis mais importantes para a personalização dos conteúdos e aplicações móveis.

Muitas aplicações baseadas em localização podem ser mencionadas, estando entre as mais promissoras os serviços de atendimento a emergências (HARGRAVE, 2000) (FCC, 2001); os de informação que disponibilizam uma central de utilidade pública oferecendo serviços como páginas amarelas, acontecimentos, eventos e atrações; os serviços de navegação (WEBRASKA, 2005) e de apoio (BOONDAO, 2003).

O crescimento no número de sistemas LBS e a diversificação imposta pelo mercado a essas aplicações aumentam de forma significativa os requisitos de *software*. Dessa forma, o contínuo incremento nos requisitos de *software* e na complexidade dos serviços oferecidos pelas aplicações LBS poderá levar a uma degradação na qualidade dos sistemas desenvolvidos, assim como à elevação nos custos e tempo de desenvolvimento e a uma conseqüente incapacidade de atender às solicitações e às constantes mudanças do mercado. Diante desse quadro, acreditamos que a solução para evitar tais problemas seria promover de forma efetiva o desenvolvimento de sistemas centrados nos preceitos da reutilização de *software*.

O desenvolvimento das diversas classes de aplicações LBS se baseia em um componente comum: a obtenção e o gerenciamento da informação de localização. Inúmeras são as possibilidades de oferecer serviços de valor adicionado e personalizados com base nas

informações de localização presentes, passadas e até futuras de objetos do nosso interesse. Porém, uma série de medidas devem ser adotadas para que essas informações possam ser utilizadas de forma funcional. Em função dessa constatação, verifica-se a necessidade de reutilização destas funcionalidades como alternativa para agilizar o processo de desenvolvimento das mesmas. Faz-se necessário, então, o desenvolvimento de um *framework* que possibilite a reutilização das funcionalidades de gerenciamento da informação de localização dos objetos móveis, pois as mesmas, juntamente com as informações georeferenciadas do cenário da aplicação, formam a base para a implementação de sistemas dessa categoria.

Este trabalho possui como objetivos principais a concepção de uma arquitetura para aplicações LBS, na qual o gerenciamento da informação de localização se configura em componentes encapsulados e reutilizáveis, bem como o desenvolvimento de um *framework* para facilitar a construção de aplicações baseadas em localização. Mais especificamente, propõe-se o desenvolvimento do *framework* denominado FRAGIL (FRAmework para Gerenciamento de Informações de Localização) para disponibilizar as funcionalidades de gerenciamento da informação de localização dos objetos móveis.

De forma mais detalhada, pretende-se investigar as arquiteturas propostas na literatura para o desenvolvimento de aplicações baseadas na localização; propor e definir uma arquitetura base para a implementação de LBS; desenvolver um *framework*, baseado na arquitetura proposta, para aplicações LBS e implementar um sistema baseado em localização que utilize o *framework* proposto de forma que se possa testar e avaliar sua funcionalidade.

O presente trabalho encontra-se organizado em mais quatro capítulos descritos a seguir.

No segundo capítulo serão apresentados o estado da arte de LBS, bem como sua conceituação e taxonomia; serão mostrados os diversos tipos de tecnologias de localização e

as classes de aplicações para LBS. Ao final, serão feitos levantamentos sobre os requisitos necessários para a construção dessas aplicações.

No capítulo terceiro alguns trabalhos relevantes serão analisados, tendo como enfoque o estudo das arquiteturas e *frameworks* para aplicações LBS propostos na literatura, ressaltando seu funcionamento, suas qualidades e possíveis limitações.

O capítulo seguinte apresentará a definição da arquitetura a ser utilizada, as funcionalidades oferecidas pelo *framework* proposto, sua modelagem e implementação.

No capítulo quinto será feita a descrição da construção de uma aplicação para validar o *framework* FRAGIL, que consiste em um sistema para o rastreamento dos usuários que estiverem transitando na região do centro histórico da cidade de São Luís, disponibilizando recursos de detecção contínua da localização, determinação de proximidade e consultas de rotas percorridas pelo usuário.

Finalmente, no sexto capítulo, serão apresentadas as considerações finais a respeito do trabalho, destacando a importância, as limitações e possíveis melhorias a serem acrescentadas.

2 SISTEMAS BASEADOS EM LOCALIZAÇÃO

Sistemas ou serviços baseados em localização (LBS – *Location-based Systems or Services*) permitem aos usuários, através de uma rede de comunicação móvel, utilizarem um conjunto de funcionalidades baseadas na sua posição ou localização geográfica. Segundo Shiode *et al*. (2002), LBS podem ser vistos como a convergência de várias novas tecnologias (NICTs – *New Information Communication Technologies*), tais como: sistemas de comunicação móvel, tecnologias de localização e dispositivos portáteis, com a Internet, Sistemas de Informação Geográfica (SIGs) e banco de dados espaciais, como mostrado na Figura 2.1.

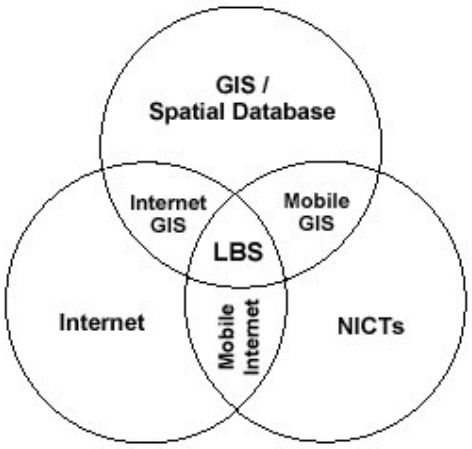

Figura 2.1. LBS como Convergência de várias Tecnologias (SHIODE *et al*., 2002)

Uma das principais características de sistemas LBS está na mobilidade de seus clientes, geralmente compostos por dispositivos portáteis (*laptops*, PDAs e telefones celulares). A capacidade de conhecer, a todo o momento, a localização de pessoas, objetos e fenômenos com aplicações que são sensíveis à posição do usuário e que podem ser facilmente acessadas através de uma grande gama de dispositivos via Internet e rede sem fio, conduz a

incontáveis benefícios para diversos setores da sociedade, tais como o comércio, os consumidores e áreas do governo.

O ponto central desses sistemas está em obter a informação necessária para determinar a localização do usuário móvel e, baseado nessa informação, oferecer serviços sensíveis a esse contexto de utilização do sistema.

De acordo com a especificação OpenLS[1] (OGC, 2004), a localização representa uma posição no espaço e no tempo, que pode ser mensurada tendo como base suas coordenadas em um sistema de referência espacial e temporal particular. Para desenvolvermos sistemas sensíveis à localização de um usuário, primeiramente, são necessárias tecnologias para determinar a localização dos clientes e processar a informação de localização.

Levando-se em consideração a forma como a localização do usuário é obtida, sugere-se a existência de três gerações de LBS (DIBDIN, 2001). A primeira geração caracteriza-se pelo fato de que o usuário deveria fornecer, manualmente, para a aplicação, a informação de onde está localizado. A entrada desses dados poderia ser em forma de um código postal (CEP) e endereços determinados por nome de cidades, ruas, quadras. Na segunda geração, a localização do dispositivo passa a ser determinada automaticamente. Por fim, na terceira e atual geração das aplicações LBS, a informação de localização é determinada automaticamente e com um alto grau de precisão, mas a principal característica que a difere da segunda geração é a proatividade, isto é, a possibilidade de oferecer um serviço sem a necessidade de o usuário solicitá-lo de forma explícita.

Segundo Deshpande e Borriello (2002), o estado dos sistemas baseados na localização é marcado pela grande gama de tecnologias de localização de posição, pela necessidade do uso de diferentes tecnologias de localização para diferentes ambientes e,

[1] É uma iniciativa do consórcio OpenGeospatial, define um conjunto de interfaces para o desenvolvimento de serviços baseados em localização, todos utilizando protocolos padrão Web.

conseqüentemente, a falta da chamada tecnologia de localização ideal, isto é, a falta de uma tecnologia que atenda a todos os ambientes e usos de maneira ótima.

2.1 Arquitetura de Sistemas Baseados na Localização

Para desenvolver sistemas sensíveis à localização de um usuário, deve existir, de uma forma geral, uma estrutura básica de componentes que dêem suporte à localização, solicitações e consultas e ao gerenciamento da informação geográfica e contextual. Primeiramente, são necessárias tecnologias para determinar a localização dos clientes e processar a informação de localização.

Como os usuários alternam entre momentos de constante movimentação e períodos estáticos, necessitamos localizá-los de forma que no instante em que uma solicitação/consulta for realizada, saibamos quais os dados devem ser enviados, levando-se em consideração a posição atual do cliente. Também é necessária uma estrutura que receba e processe as solicitações dos clientes e envie a informação pedida para o mesmo através de uma rede sem fio e no formato correto, de forma que a informação possa ser mostrada apropriadamente no dispositivo do usuário. Na Figura 2.2 pode-se observar uma arquitetura genérica para sistemas baseados na localização do usuário.

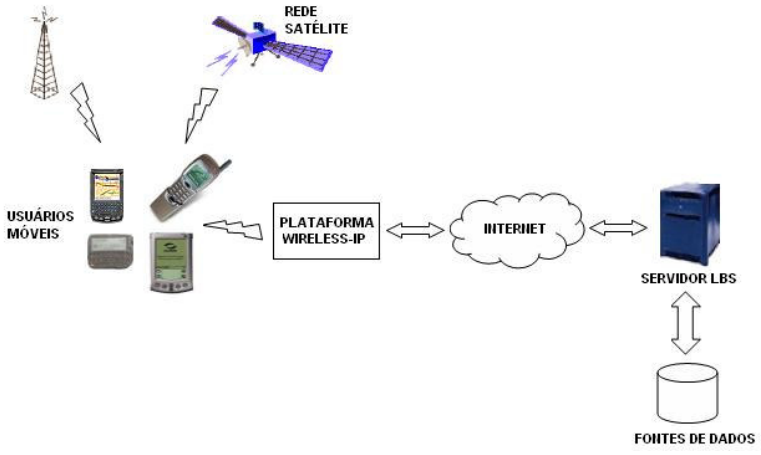

Figura 2.2 Arquitetura Geral de um sistema LBS

Entre os componentes necessários para a construção de aplicações LBS, podemos destacar: componentes para a determinação da posição, aquisição da informação sobre a localização, serviços de processamento da localização e o serviço de *Gateway*.

- Determinação da posição

Os componentes de determinação da posição são responsáveis em obter a posição dos dispositivos móveis em tempo real, consistindo em *hardware* e *software* para identificar e rastrear a localização dos usuários. O *hardware* é composto basicamente pelo equipamento de determinação da posição (PDE – *Position-Determining Equipment*) que identifica a localização dos dispositivos móveis (MAPINFO, 2002). Existem basicamente duas formas para obter a posição de um dispositivo móvel, são elas: soluções baseadas em rede (*network-based*) e soluções baseadas no dispositivo (*handset-based*). Na seção 2.3 abordaremos com maiores detalhes essas tecnologias de localização.

Além de determinar a posição do dispositivo móvel, faz-se necessário a existência de um componente que processe, rastreie e administre a informação de localização (PENG, 2003). Esse componente pode estar localizado no servidor da rede móvel à qual o PDE está conectado (rede celular ou rede sem fio) ou em um outro servidor externo à rede. O objetivo desse componente é ajudar outras aplicações a consultar ou recuperar a informação de localização, que pode ser fornecida em tempo real ou armazenada no servidor para consultas temporais.

- Aquisição da informação de localização

A informação sobre a localização se refere tanto ao conteúdo que descreve características relacionadas a uma posição geográfica quanto à própria posição geográfica.

Nesse contexto, um sistema de localização pode fornecer vários tipos de informação: física ou simbólica e absoluta ou relativa. Entende-se por física a localização geográfica ocupada pelo usuário. Por exemplo, o GPS oferece posições físicas: um prédio está situado a 47°39'17'' N, 122° 18' 23'' W e 20,7 metros de altitude. Em contrapartida, a localização simbólica aborda um posicionamento abstrato de onde alguma coisa está localizada como, por exemplo, na cozinha, na próxima rua, ao lado do banheiro.

Sistemas que oferecem posicionamento físico podem ser estendidos para suportar uma localização simbólica correspondente e utilizar o posicionamento físico para determinar um conjunto de informações simbólicas. A informação absoluta possui uma matriz de referência única para especificar a localização de um determinado objeto, expressa, globalmente, em latitude, longitude e altitude. Adicionalmente, pode-se ter um atributo de direção complementando a localização absoluta. Já no sistema relativo, cada objeto pode ter a sua própria matriz de referência (HIGHTOWER; BORRIELLO, 2001).

Tanto a informação geográfica de usuários e pontos/áreas de interesse quanto as informações que caracterizam e descrevem essas posições são fundamentais para o perfeito funcionamento de uma aplicação LBS. A informação geográfica mostra onde algo está posicionado, o que ou quem está em determinado local. Por outro lado, existem aquelas informações que descrevem atributos de uma localização em particular, inserindo um ponto geográfico em um contexto conhecido pela aplicação e pelos usuários. Essas informações incluem escolas, lojas, hospitais, órgãos governamentais, bem como informações detalhadas sobre qualquer local de interesse do usuário como, por exemplo, o cardápio de um restaurante, a lista de filmes em cartaz no cinema, os principais itens em promoção em uma loja, entre outros.

A existência de componentes que mantenham a atualização e a precisão dessas informações é de fundamental importância para as aplicações LBS.

– Serviços de processamento da localização

Esses componentes devem realizar processamentos e disponibilizar serviços baseados na informação geográfica, de forma a satisfazer as consultas dos usuários oferecendo uma visão amigável e contextualizada dessas informações. Alguns serviços que podem ser oferecidos por esses componentes são: serviços de consulta e visualização, serviços de geocodificação (procurar a localização física a partir de um endereço específico) e geocodificação reversa, serviços de planejamento de rotas e serviços de geração e visualização de mapas (transformação da informação de localização em mapas e imagens).

– Serviço de *Gateway*

É um *midleware* entre os serviços de processamento e os dispositivos móveis. Tem como objetivo fazer com que os serviços oferecidos pelo servidor sejam compatíveis com o dispositivo do usuário. Assim, o papel do *gateway* inclui: obter a atual posição de um objeto móvel a partir de um servidor de localização; receber e processar as solicitações advindas do usuário e encaminhá-las para um servidor *Web* ou outra aplicação e, por fim, converter as respostas em um formato que possa ser visualizada, levando-se em consideração o sistema operacional e as limitações dos diferentes dispositivos móveis.

Além dos componentes relacionados anteriormente, temos que os sistemas LBS podem depender tanto da comunicação através de rede com fio quanto das redes sem fio. Também necessitam que os dispositivos portáteis sejam capazes de acessar e visualizar serviços através da rede sem fio.

2.2 Requisitos de Aplicações Baseadas em Localização

Diversas são as classes de aplicações que podemos desenvolver utilizando como característica principal o contexto de localização do usuário. Porém, pode ser identificado, de um modo geral, um conjunto de funcionalidades que seriam essenciais em diversas aplicações LBS. A seguir, serão enumeradas algumas delas:

a) definir Pontos de Interesse (POI) de forma individual ou classificados em categorias;
b) criar um perfil (uma espécie de favoritos) em que o usuário possa adicionar uma referência a um POI para uso futuro;
c) destacar, graficamente, em mapas, POIs próximos ao usuário;
d) permitir que o usuário defina seus próprios pontos de interesse, adicionando conteúdo através de textos e imagens;
e) permitir que o usuário edite o conteúdo relacionado a um POI existente;
f) visualizar o mapa de uma área de interesse (AOI);
g) facilidades de roteamento: fornecer uma seqüência de instruções indicando a direção correta até um POI e mostrar graficamente o caminho para um POI ou para um conjunto de POIs;
h) permitir consultas baseadas no menor caminho;
i) visualizar a posição do cliente: no dispositivo do cliente, em uma central de monitoramento (vários clientes) e até mesmo em outros clientes;
j) disponibilizar serviços do tipo *pull*: neste tipo de serviço a solicitação para a realização de uma consulta/tarefa é feita de forma explícita pelo usuário;
k) disponibilizar serviços do tipo *push*: estes são exatamente o oposto dos serviços do tipo *pull*, aqui de forma autônoma e proativa podem realizar tarefas

e enviá-las ao cliente móvel, baseando-se na ocorrência de eventos ou condições satisfeitas. Enviar mensagens promocionais baseadas no interesse do usuário e na sua posição e receber mensagens quando estiver próximo a outros usuários de seu interesse são exemplos de serviços *push*.

2.3 Tecnologias de localização de dispositivos móveis

A localização de um usuário em uma rede sem fio é muito mais complexa do que localizar um usuário que utiliza um meio fixo como uma linha de telefone para comunicação. O usuário de rede sem fio pode estar em qualquer ponto de uma certa área de serviço. Nesse caso, tanto a estação móvel quanto a rede que a compõe devem ter a capacidade para localizar com precisão o usuário e, uma vez que este pode se mover de uma área para outra, a atualização da sua localização também se torna necessária. Um outro ponto que dificulta a localização desses usuários é o fato de esse tipo de rede operar em ambientes onde existe abundância de ruídos, interferências e desaparecimento do sinal a localizar. Outro ponto que adiciona complexidade ao processo é a existência de múltiplas tecnologias de rede (TDMA, CDMA, GSM, IEE802.11, etc.), muitas vezes com interfaces completamente incompatíveis.

Atualmente existem diversas tecnologias que podem fornecer a localização de um usuário móvel, sendo que cada uma delas possui vantagens e desvantagens específicas. Apesar da grande quantidade, ainda não existe um sistema de posicionamento ideal, isto é, aquele que atenda de forma precisa e eficaz a todas as categorias de LBS. Sistemas de posicionamento baseados em satélites, como GPS, obtêm alto alcance e precisão, porém falham em ambientes fechados. Por outro lado, sistemas ditos *indoor* necessitam de elevados custos de instalação e são restritos a prédios ou algumas áreas específicas dentro de uma construção.

Sistemas baseados em localização, através de um PDE, devem ser capazes de detectar a posição de um usuário móvel. A escolha da tecnologia a ser utilizada deverá ser baseada em uma combinação de fatores que incluem o grau de precisão, a área de abrangência (cobertura), os custos e a natureza do serviço a ser oferecido.

As tecnologias utilizadas para a localização de dispositivos móveis podem ser classificadas de acordo com o local onde as coordenadas dos dispositivos são coletadas e calculadas, podendo ser classificadas em: soluções baseadas em rede (*network-base*d), soluções baseadas no dispositivo (*handset-based*), soluções híbridas. Adicionalmente, podemos identificar como um caso especial das soluções baseadas em rede as soluções para ambientes fechados (*indoor*).

2.3.1 Soluções baseadas em rede *(network-based)*

As soluções baseadas em rede se caracterizam pela presença de um equipamento de localização colocado nas estações rádio-base[2] (ERB's) do sistema. Assim, todo o processamento matemático necessário para a determinação das coordenadas dos dispositivos é feito na infra-estrutura da própria rede.

2.3.1.1 Identificação de Célula (*Cell Identification - CID*)

Também conhecida como *Cell of Global Identity* (CGI), é considerada a técnica mais simples de ser implementada, baseia-se no fato de que a localização da estação rádio-base (ERB) determina a localização do dispositivo móvel. Uma vez que a ERB para cada

[2] Estação Rádio Base (ERB) ou "Cell site" é a denominação dada, em um sistema de telefonia celular, para a Estação Fixa que contém um conjunto de equipamentos de telecomunicações e eletrônicos que são conectados a um ou mais sistemas irradiantes (antenas), com a finalidade de criar uma área de cobertura (célula).

célula possui uma localização fixa, facilmente temos a posição dos objetos que estão situados dentro da referida célula (HJELM, 2002).

O ponto fraco dessa abordagem é que não é possível determinar a exata posição de um objeto na área de cobertura da ERB. Portanto, a precisão de localização depende do tamanho da célula, podendo variar de 150 m (em uma "micro-célula") até 20 km (para uma "macro-célula"), o que pode ser aceitável para ter uma idéia geral de onde o usuário está localizado, não fornecendo informações satisfatórias para aplicações de emergências ou de rastreamento. Tal abordagem caracteriza-se por ser muito rápida e facilmente disponível, já que depende apenas da identificação da estação rádio-base responsável pela solicitação do serviço.

Algumas técnicas podem ser aplicadas ao CGI para tentar incrementar a precisão oferecida. Um exemplo seria a abordagem onde as células são divididas em secções, reduzindo a área total da provável localização. As estações rádio-base possuem várias antenas (geralmente 2, 3 ou 4) que dividem o espaço de abrangência de 360 graus em diversos segmentos angulares de 180, 120 ou 90 respectivamente, limitando, assim, a área onde o usuário móvel possa ser localizado. Por exemplo, se a área de cobertura da antena é de 4 km², a localização do usuário fica delimitada por essa área. Se, no entanto, essa célula puder ser identificada em seções de 120 graus, então a região onde o usuário está localizado reduz-se a um terço da área original (MALLICK, 2003).

2.3.1.2 Ângulo de Chegada (*Angle of Arrival* -AOA)

Esta abordagem ao problema da localização se dá com base no cálculo do ângulo com que determinado sinal chega às antenas da rede celular (Figura 2.3). 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 desse grupo tem como resultado os ângulos de incidência, podendo ser referenciado em relação a qualquer direção fixa (MALLICK, 2003). Para obtermos a informação necessária para a localização da fonte do sinal, basta termos os ângulos de incidência do sinal em apenas dois receptores. O aumento da precisão de estimação da fonte ocorre à medida que se eleva a quantidade de antenas leitoras do sinal em uma determinada área.

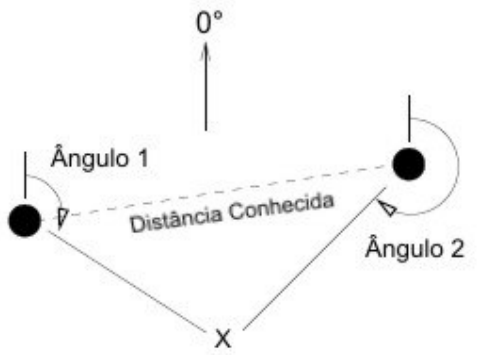

Figura 2.3 Técnica de Angulação

O AOA é um método relativamente simples, uma vez que aproveita os sinais transmitidos pelos aparelhos móveis, não requerendo modificações aos aparelhos. Contudo, um alto custo de implantação é necessário em virtude da necessidade de utilização de antenas especias na rede de comunicações. Outras vantagens incluem a necessidade de apenas dois receptores para determinar a posição de um sinal, sendo necessário, apenas, se mais informações estiverem disponíveis, como a intensidade do sinal. Alta precisão pode ser obtida se as estações estiverem a menos de 8 km de distância umas das outras. A largura de banda necessária é mínima e o atual desenvolvimento de pequenas antenas utilizadas para captar o sinal dá uma boa demanda na utilização dessa técnica de localização.

2.3.1.3 Tempo de Chegada (*Time of Arrival* -TOA)

Nesta abordagem, a distância do transmissor até à antena base pode ser calculada através da interpretação do tempo da propagação de um sinal em cada receptor (Figura 2.4). Alternativamente pode ser calculada a distância entre transmissores e um receptor de sinal. O TOA de cada sinal é comparado com o (conhecido) tempo de transmissão de cada transmissor. É necessária uma sincronização bastante precisa para que os resultados possam localizar com precisão o aparelho transmissor de sinais.

A maior parte dos sinais utilizados é de forma conhecida e são codificados com largura de banda suficiente para que a recepção de três ou mais desses sinais possa determinar a localização do receptor de modo síncrono. Uma grande desvantagem dessa técnica está no alto custo de implementação devido à exigência de temporizadores precisos em cada estação base. O grau de precisão dessa técnica é considerado aceitável, obtendo aproximadamente 50 m para áreas urbanas e 150 m para áreas rurais. TOA é mais comum em rede CDMA/CDMA2000, já que estas já se apresentam com sincronização, não sendo necessário o uso de temporizadores adicionais (MALLICK, 2003).

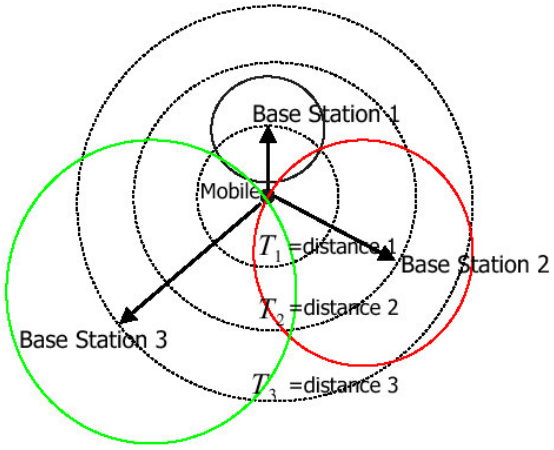

Figura 2.4. Tempo de Chegada (NACIF, 2005)

2.3.1.4 Diferença do Tempo de Chegada (*Time Difference of Arrival* -TDOA)

Os sistemas que possuem como método o TDOA utilizam transmissores de radiofreqüência (TRF) no dispositivo móvel e receptores são instalados em várias antenas utilizadas como estações rádio-base. O dispositivo móvel, através do TRF, emite sinais que são modulados com largura de banda suficiente de modo a fornecer uma alta precisão temporal (inversamente proporcional à largura de banda), obtendo, assim, uma boa precisão na estimativa da localização em questão. Partindo-se do princípio de que o sinal trafega a velocidade constante da luz, a diferença do tempo de chegada de sinal em um par de células pode ser usada para computar posições hiperbólicas em volta do dispositivo móvel. A interseção das hipérboles é então determinada para apontar a posição do celular. Nesta abordagem, a diferença de tempos de chegada do sinal a diferentes locais pode estimar diferenças de alcance. Quando três ou mais estimativas de diferentes locais estão disponíveis, a localização pode então ser calculada.

A TDOA requer a construção de uma infra-estrutura separada de antenas emissoras ou receptoras e/ou a instalação de transmissores em cada dispositivo móvel. A precisão desse tipo de tecnologia é determinada pelas limitações de largura de banda do sinal, bem como pela precisão do sistema e pelo ambiente em que o sinal se propaga (à imagem de tecnologias anteriores). Outras desvantagens incluem a extrema precisão temporal necessária em cada receptor, a necessidade de recalibrar esses mesmos receptores devido a diferentes influências no sinal, e a necessidade de enviar a informação do TOA (bem como o padrão do sinal) para um processador central. A principal vantagem da TDOA é o fato de não ser estritamente necessário modificar os dispositivos móveis, a precisão não depende da distância do transmissor, sendo aproximadamente de 125 metros.

2.3.1.5 Padrão de Trajetória (*Multipath Fingerprinting*)

Para localizar uma unidade móvel, essa técnica se baseia na comparação do padrão da trajetória percorrida do dispositivo móvel à estação base com um conjunto de padrões armazenados numa base de dados dos padrões de todas as freqüências em várias localizações. Um padrão de trajetória é o resultado da colisão dos sinais enviados com os vários obstáculos presentes durante o percurso até a estação receptora. Esses caminhos são coletados pelo sistema formando uma única "impressão digital de localização".

Nesta técnica não é necessária a existência de uma linha de visão direta entre o dispositivo móvel e as estações receptoras, uma vez que seu princípio de funcionamento está baseado nas reflexões multi-caminhos. Uma grande desvantagem é que esta abordagem não é adequada para ambientes muito dinâmicos, já que as modificações na topologia deverão ser sempre acompanhadas pela criação de um novo padrão para esses locais. Assim, para proporcionar uma localização precisa do usuário, faz-se necessário a existência de uma rica base de dados de padrões de freqüência e de uma constante atualização de seus registros (BOERTIEN; MIDDELKOOP, 2002).

Essa solução se adapta à infra-estrutura atual das operadoras. Somente uma ERB é requerida para receber o sinal e comparar com a base de dados e os dispositivos não necessitam sofrer alterações.

2.3.2 Soluções Baseadas no Dispositivo (*Handset-Based*)

A seguir serão descritas soluções baseadas nos dispositivos que, por sua vez, implementam grande parte da tecnologia nas unidades móveis e que utilizam como base o uso da tecnologia GPS.

2.3.2.1 GPS (*Global Positioning System*)

O Sistema de Posicionamento Global foi concebido pelo Departamento de Defesa dos EUA no início da década de 60, sob o nome de projeto NAVSTAR (*Navigation System with Timing and Ranging*). O sistema foi declarado totalmente operacional apenas em 1995. Esse sistema é composto por 24 satélites que se movem em seis diferentes órbitas conhecidas em volta do planeta, sendo quatro satélites por órbita (Figura 2.5). Tais satélites fornecem informação posicional 24 horas por dia, enquanto que os relógios atômicos que os satélites possuem no seu interior têm a precisão de não se atrasarem mais de 1 segundo em 30 anos. O sistema GPS é constituído, de uma forma geral, pelos satélites no espaço (*space segment*), pelas estações que monitoram e controlam esses mesmos satélites (*control segment*) e pelas unidades receptoras (*user segment*) (IGEB, 2001).

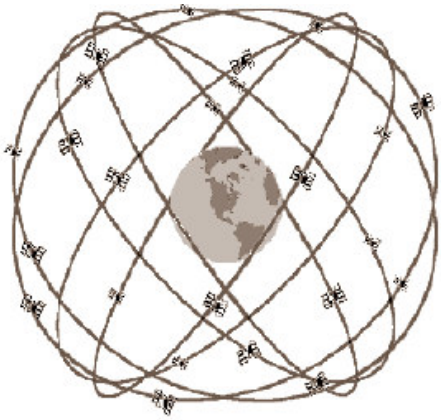

Figura 2.5 Rede satélite – GPS (SCHILLER, 2004)

O GPS funciona com base no princípio da triangulação. Conhecendo a distância em relação a três ou mais satélites, o receptor pode calcular a sua posição com base numa série de equações. Em teoria, a distância pode ser calculada multiplicando o tempo que o sinal demora a chegar pela velocidade que este viaja (a velocidade da luz, aproximadamente

300000 km/s). No entanto, na prática são necessários cálculos mais sofisticados, uma vez que podem existir inúmeras interferências que influenciam diretamente na precisão do sistema. Vejamos quais fatores podem interferir para a distorção da informação de localização em sistemas GPS:

a) flutuações na órbita: as forças gravitacionais do sol e da lua fazem com que os satélites não se movam exatamente na órbita que havia sido calculada, ocasionando erros de aproximadamente 2,5 m;
b) perturbações atmosféricas: a pressão atmosférica e as condições climáticas causam distorções no sinal de comunicação dos satélites gerando erros em torno de 0,5 m;
c) perturbações na ionosfera: o acúmulo de partículas na ionosfera podem corromper os sinais de comunicação dos satélites gerando erros de até 5,0 m;
d) múltiplos caminhos: a reflexão de sinais no ambiente do receptor causa erro de 0,6 m.

Para calcular a longitude e a latitude é preciso que sejam obtidas informações de três satélites diferentes. No entanto, são necessários quatro para calcular também a altitude.

Preocupados com o uso inadequado da tecnologia, os militares americanos implantaram dois serviços com diferentes opções de precisão: *Precision Positioning Service* (PPS) e a *Standard Positioning Service* (SPS). O PPS garante em 95% do tempo uma precisão de 22 m na horizontal e 27,7 m na vertical. O sinal desse serviço é criptografado e somente pode ser decodificado pelas forças armadas do governo norte-americano. Já o SPS é destinado a usuários não-autorizados (civis) e disponibilizado para o uso comercial. O sinal fornecido por esse serviço é distorcido artificialmente para efetivar uma informação mais exata a respeito da localização de um receptor. Tal mecanismo é conhecido como *Selective Availability* (SA). Até o ano 2000 o SPS permitia uma precisão de 100 m na horizontal e 156 m na vertical. Em maio de 2000, o então presidente dos Estados Unidos, Bill Clinton decretou

o fim do SA (OSTP, 2000), disponibilizando para os usuários civis a precisão média de 10 m na horizontal e 23 m na vertical (IGEB, 2001).

Em uma análise geral, o GPS apresenta-se como uma tecnologia de localização muito precisa, funcionando, a princípio, em qualquer lugar do planeta e sofrendo influências mínimas das condições ambientais no processo de posicionamento. Porém, algumas desvantagens podem ser apresentadas, como o custo para manutenção e supervisão da rede de satélites e o fato de que a posição só pode ser determinada quando se recebe o sinal de pelo menos três satélites, tornando praticamente impossível o uso em ambientes fechados (*indoor*).

2.3.2.2 AGPS (*Assisted-GPS*)

Assisted-GPS (AGPS) é uma tecnologia que combina o uso do GPS com a infra-estrutura de rede para obter melhores resultados (DJUKNIC; RICHTON, 2001). O sistema é composto, principalmente, pelos dispositivos móveis com um receptor GPS parcial, por uma rede de servidores GPS estacionários (servidores AGPS) que tem a característica de "visualizar" os mesmos satélites do dispositivo e pela infra-estrutura de rede constituída pelas estações rádio-base (ERB) e pelo MSC[3] (*mobile switching center*). O princípio básico dessa técnica está centrado na distribuição das funções de detecção do posicionamento entre os dispositivos móveis e os servidores da rede.

Os servidores AGPS podem obter a localização (com precisão em nível de célula ou setor) dos dispositivos móveis a partir do MSC e passar a monitorar os mesmos sinais dos satélites GPS detectados pela estação móvel. Assim, a rede pode de forma precisa, prever os sinais GPS que o dispositivo móvel iria receber e enviar essa informação para o mesmo,

[3] É a parte central de um sistema celular, responsável pela validação dos assinantes, processamento de chamadas, interface com a rede fixa de telefonia, interface com outros MSC´s sejam eles de outra operadora ou não, geração de bilhetes das chamadas, gerenciamento de hand-off (passagem do móvel de uma célula para outra), monitoração de alarmes das Estações Radio Base – ERBs ,entre muitas outras funções.

reduzindo de forma significativa as funções de troca de informações antes desempenhadas pelo dispositivo, o tamanho do espaço de busca e conseqüente diminuição no TTFF de minutos para segundos (DJUKNIC; RICHTON, 2001).

A distribuição de dados e processamento entre a rede e os dispositivos móveis otimiza o tráfego pela rede sem fio, economiza recursos de energia, simplifica os requisitos do receptor GPS e possibilita a detecção de sinais mais fracos do que no sistema GPS tradicional. A precisão alcançada pelo AGPS é de aproximadamente 50 metros para ambientes fechados e de 10 metros para áreas livres.

2.3.2.3 E-CID (*Enhanced Cell ID*)

A tecnologia E-CID é baseada em *software* e usa a arquitetura da rede GSM e dos *handsets*. Ela não possui os empecilhos comuns de outras tecnologias, como ambientes *indoor* e a linha de visão. Em cada dispositivo móvel existe uma tabela constantemente atualizada que possui as ERB's possivelmente conectáveis, permitindo que no decorrer do trajeto use outra ERB na transmissão de dados ou voz. Para obtenção do posicionamento, o servidor da rede faz um cálculo de triangulação e determina a área de interseção das células atingidas por ele. A precisão dessa solução aumenta quando a concentração de células aumentarem (ZURSTRASSEN, 2005).

A implantação da E-CID apresenta um baixo custo, uma vez que não implica em modificações nos dispositivos móveis, nem tão pouco na estrutura atual das redes. Outra vantagem desta abordagem é que não é necessário que o usuário estabeleça uma linha de visão direta com as antenas receptoras, assim pode ser perfeitamente utilizada em ambientes com grande concentração de construções. O fato de a precisão da localização ser diretamente proporcional à concentração de células, é apontada como sua grande desvantagem. Desta

forma, em regiões metropolitanas ela pode chegar a 50 m, no entanto em áreas rurais pode vir a chegar a mais de 2 km.

2.3.3 Soluções Híbridas

Uma solução híbrida envolve combinação e interação entre a tecnologia do dispositivo móvel e a rede de telefonia para determinar a localização. Um exemplo de solução híbrida é o método E-OTD *(Enhanced Observed Time Difference)* que se baseia no cálculo do tempo de propagação do sinal. O princípio geral da E-OTD baseia-se na recepção dos sinais das várias estações bases circunvizinhas por um dispositivo móvel e na computação das diferenças de tempo para que esses sinais alcancem o dispositivo comparado aos receptores fixos da rede (Figura 2.6).

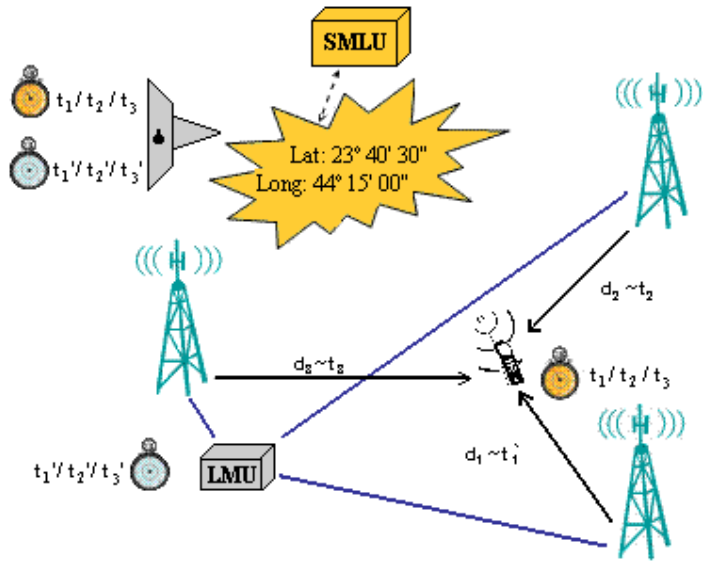

Figura 2.6 Enhance Observed Time Difference (E-OTD)

Para a transmissão apropriada dos *timeslots* o dispositivo móvel deve ser sincronizado com a estação base (ERB). Entretanto, a rede GSM não é sincronizada de forma

global. O problema da sincronização é resolvido com a ERB emitindo uma "mensagem de sincronização" em intervalos regulares. O dispositivo móvel monitora estas mensagens enviadas pelas ERBs mais próximas e ajusta seu sincronismo adequadamente.

No sistema E-OTD o telefone móvel escuta as mensagens das ERBs vizinhas e grava os diferentes tempos em que as mesmas são recebidas. O receptor fixo na rede, chamado de Unidade de Medida da Posição (*Location Measurement Unit* - LMU) também escuta as mensagens de sincronização e determina o tempo em que cada mensagem foi transmitida. Dado os tempos de recepção e os conhecidos tempos de transmissão, a rede calcula a posição do dispositivo móvel. Este método apresenta precisão que varia de 250m a 100m, tendo um melhor desempenho em áreas com grande concentração (densidade) de ERBs, inclusive em ambientes fechados.

2.3.4 Sistemas de posicionamento para ambientes fechados (*indoor*)

Os sistemas baseados na rede satélite, como GPS, apresentam alto alcance e precisão, mas falham em ambientes fechados. Quando existe a necessidade de obter a localização de objetos e ou pessoas localizadas em um prédio, deverá ser utilizada uma das técnicas baseadas em sensores mostradas a seguir.

2.3.4.1 Localização Baseada em raios infravermelhos

Uma grande classe de sistemas se utiliza dessa técnica para obter a localização de um usuário em ambientes fechados, como prédios e construções de um modo geral. Ela é constituída basicamente por um conjunto de dispositivos receptores e emissores de sinais infravermelhos. A cada intervalo de tempo, um sinal é gerado pelo emissor. Como a

localização dos receptores é conhecida, ao sabermos qual receptor capturou o sinal infravermelho, estaremos conhecendo o lugar onde o objeto está localizado.

Um grande representante dessa classe de localização é o *Active Badge System* (WANT *et al*., 1992). Este foi desenvolvido pelo *Olivetti Research Laboratory* e consiste em um sistema de localização baseado no uso da tecnologia de difusão de raios infravermelhos. O sistema é composto por dispositivos emissores de raios infravermelhos, cada um associado a uma pessoa ou objeto a localizar, denominados *Active Badges*, bem como em uma rede de dispositivos receptores de raios infravermelhos associados a uma localização, denominados sensores (*beacons*).

Cada indivíduo usando um *Active Badge* transmite um sinal de raios infravermelhos a cada 10 segundos, que é detectado pelos sensores. O sinal transmitido pelo *Active Badge* consiste em um código de identificação (ID – único para cada dispositivo) e na informação do status do *Badge*. Essa informação, detectada pelos sensores, é enviada para um servidor que faz a gestão da rede de sensores e a interface entre esta e as aplicações. Esse servidor periodicamente pesquisa os sensores para recolher a informação transmitida pelos *Active Badges* e transfere essa informação para a rede *Ethernet*. Tal informação é recolhida pelos clientes e associada com a informação contida em uma base de dados para posterior processamento.

Os *Active Badges* possuem também capacidades receptoras que, associadas a um pequeno *dispositivo* que têm incorporado, permitem às aplicações-cliente enviar diferentes sinais que serão interpretados de acordo com algum protocolo estabelecido na organização. Além disso, trazem incorporados dois botões cuja funcionalidade não está estabelecida, podendo ser programados *ad hoc*, segundo as necessidades dos utilizadores. Essas duas últimas características permitem uma comunicação bidirecional entre as aplicações cliente e os portadores dos *Active Badges*.

O sistema em questão fornece uma informação de localização simbólica, representando os espaços físicos no qual o sensor está instalado. Sua precisão limita-se a esses espaços, fornecendo informações como, por exemplo, o usuário de identificação x está na sala de número y. Esse sistema, além de informar a simples localização de seus usuários, pode ser utilizado para uma série de outros propósitos, tais como: controle de acesso a determinadas áreas, controle de determinados dispositivos de forma automática (abertura e fechamento de portas, iluminação automática de áreas, ventilação, redirecionamento de chamadas telefônicas), ajuda ao resgate de pessoas em caso de emergência médica, guia e controle de visitantes, gestão de eventos comerciais ou conferências, etc.

2.3.4.2 Localização Baseada em Radiofreqüência

Vimos, anteriormente, que devido à natureza dos raios infravermelhos, as técnicas de localização que se utilizam dessa tecnologia oferecem apenas informações de localização semântica, limitando a precisão da posição ao espaço físico de uma sala, por exemplo. Os sinais de rádio, ao contrário, podem ultrapassar paredes. Dessa forma, se vários rádio-transmissores distribuídos por um prédio são "visíveis" ao objeto a ser localizado, então uma posição física do mesmo poderá ser calculada. Adicionalmente, se transmissores são dispostos em diferentes níveis, uma posição tridimensional poderá ser determinada.

A RFID (*Radio Frequency Identification*) apresenta-se como uma variante dessa classe de sistemas de localização. Caracteriza-se como uma tecnologia de identificação que utiliza a radiofreqüência para capturar dados. A tecnologia surgiu inicialmente na década de 80 como uma solução para os sistemas de rastreamento e controles de acesso.

Em 1999, o *Massachusetts Institute of Technology* (MIT), juntamente com outros centros de pesquisa, partiram para o estudo de uma arquitetura que utilizasse os recursos das

tecnologias baseadas em radiofreqüência para servir como modelo de referência para o desenvolvimento de novas aplicações de rastreamento e localização de produtos. Desse estudo nasceu o Código Eletrônico de Produtos – EPC (*Electronic Product Code*). O EPC definiu uma arquitetura de identificação de produtos que utilizava os recursos proporcionados pelos sinais de radiofreqüência e que foi chamada posteriormente de RFID ou Identificação por Radiofreqüência (SANTOS, 2004).

Os sistemas de RFID são constituídos de leitores e "etiquetas inteligentes" (*microchips* conectados às antenas). Quando a etiqueta está posicionada próxima de um leitor, ela transmite a informação contida no seu *chip*. Como os *chips* não contêm nenhuma fonte de potência, eles só podem transmitir as informações neles contidas no máximo a uma distância de 70 cm. Para transmissão de dados, as etiquetas respondem a sinais de rádio de um transmissor, enviando de volta informações quanto à sua localização e identificação.

2.3.4.3 Localização Baseada em ultra-som

Desenvolvido no AT&T *Laboratories Cambridge*, o *Active Bat System* (AT&T, 2005) surgiu com o propósito de atender a um conjunto de aplicações que necessitam da informação de localização e orientação tridimensional e com um alto grau de precisão, características não suportadas pelo *Active Badges System*.

Um pulso de ultra-som é emitido a partir de pequenas unidades transmissoras, denominadas *Bats*, que estão presas aos objetos a serem localizados. O tempo que esse pulso leva do *Bat* até as unidades receptoras de ultra-som, localizadas em pontos conhecidos no teto das salas a serem monitoradas, é medido. Como a velocidade do som no ar é conhecida, é possível calcular a distância do *Bat* para cada receptor.

Se a distância do *Bat* para três ou mais receptores não lineares puder ser obtida, então a posição de um objeto no espaço 3D pode ser determinada usando o processo de multilateração. Pela busca das posições relativas de dois ou mais *Bats* presos a um objeto, também é possível calcular sua orientação. Reflexões dos sinais ultra-sônicos nos objetos do ambiente são comuns e podem gerar informações incorretas sobre as medidas de distância para os *Bats*. Esses erros são eliminados pelo uso de um algoritmo estatístico de rejeição para essas informações, mantendo um alto grau de precisão da localização (HARTER *et al*., 2001). Esse sistema é capaz de fornecer uma precisão na localização dos *Bats* de 9 cm da posição original para 95% das identificações (HIGHTOWER; BORRIELLO, 2001).

2.4 Aplicações de Sistemas Baseados na Localização

Como a tecnologia de localização adiciona valor ao serviço sem fio existente, os fornecedores deste estarão interessados em colocá-la no mercado. Os serviços de proteção e emergência serão os primeiros beneficiados, mas existem muitos outros campos de aplicação, como a indústria dos transportes ou mesmo estudos de mercado sobre como melhor servir a população.

O surgimento de uma nova geração de dispositivos móveis (na qual se incluem os primeiros terminais 3G), assim como a miniaturização e cada vez maior acessibilidade dos serviços de posicionamento de elevada precisão baseados em GPS, aliada à maior facilidade de programação dos dispositivos (permitindo, por exemplo, a construção de aplicações multimídia independentes da plataforma baseadas na tecnologia Java), abre as portas para uma nova geração de aplicações.

A aplicação generalizada da tecnologia de posicionamento, a partir do momento em que tal funcionalidade se torne um padrão, abre novos horizontes nas aplicações móveis.

A integração de dados do GPS com sistemas de informação geográfica ou bases de dados com informação acerca dos pontos de interesse numa determinada zona possibilitam a construção de aplicações multimídia, sugerindo rotas ou pontos de interesse, incluindo serviços como "você está aqui", "percursos recomendados", "locais e monumentos importantes a visitar" e na construção de "*browsers*" que, com recurso à informação recebida pelo sistema de posicionamento (GPS ou célula), permitem fornecer a informação necessária aos servidores *Web* no sentido da disponibilização de informações contextualizadas.

Pesquisa mostrada em Shiode *et al*.(2002) demonstra a tendência de mercado e o potencial reservado para as aplicações centradas na localização do dispositivo, que a cada ano ganha fundamental importância entre os usuários, revelando-se uma área de domínio entre as aplicações de dispositivos móveis em 2005, como mostra o Quadro 2.1.

Quadro 2.1 Tendência das aplicações utilizadas em dispositivos móveis

Rank	2000	2003	2005
1	PIM[4]	PIM	Navigation / Location
2	Entertainment	Entertainment	PIM
3	Financial Services	Navigation / Location	Entertainment
4	Internet Browsing	Financial Services	Financial Services
5	Navigation / Location	Internet Browsing	Internet Browsing
6	m-Commerce	m-Commerce	m-Commerce
7	Intranet	Intranet	Intranet

Por outro lado, os serviços de rastreamento de veículos e monitoramento de frotas por satélites, via rede de celulares, teve um aumento ainda maior na demanda. Segundo a *ABI Research*, as receitas com esse tipo de serviço atingirão US$ 5 bilhões em 2009, contra US$ 1,5 bilhão atualmente. Os números são mais conservadores do que da pesquisa da *Business Communications Company*, a qual revela que o mercado de telemática atingirá US$ 6,5

[4] Personal Information Management

bilhões já em 2008. Os analistas da *ABI* afirmam que, embora as aplicações comerciais estejam apenas começando a surgir, o mercado corporativo se firmará como o principal consumidor da tecnologia, utilizando-a, principalmente, para rastrear ativos ou empregados em campo (MOURA, 2004).

Apesar de já existir bastante trabalho em curso no âmbito da definição de modelos de negócio que permitam que tanto as operadoras quanto os autores das aplicações tirem, simultaneamente, partido de todas as oportunidades em aberto, é esperado que num futuro próximo essas aplicações possam representar uma considerável fonte de receitas para as operadoras móveis.

Em resumo, pode-se concluir que, em termos de negócio de conteúdos, a exploração da informação posicional tem o potencial de inserir o contexto geográfico do utilizador como uma das variáveis mais importantes para a personalização dos conteúdos e aplicações móveis.

As operadoras, além de serem capazes de oferecer serviços inteiramente novos aos seus clientes, poderão também oferecer aperfeiçoamentos nos serviços correntes, tais como tarifação sensível à localização ou serviços de informação. Como exemplo, pode-se oferecer serviços que disponibilizem informações a respeito da intensidade do tráfego de veículos em uma determinada região, localização e tipos de restaurantes próximos a um usuário, guias de turismo, monitoramento de carros e caminhões, sistemas de rotas otimizadas e um conjunto cada vez maior de novas aplicações que tem como fator chave a localização do usuário.

A seguir, serão apresentados, com mais detalhes, alguns sistemas baseados na localização que se destacam pelo grande potencial de utilização, tendo em vista a tecnologia disponível atualmente e a demanda do mercado por esses serviços.

2.4.1 E911

O principal fator que impulsionou os serviços de localização nos Estados Unidos foi a criação do mandato E911 por parte do *Federal Communications Commission's* (FCC). O grande aumento no número de ligações originadas de unidades móveis ao serviço de emergência americano passou a ser um empecilho ao serviço de segurança pública, já que não era possível identificar o local onde as pessoas estavam, a não ser que essa informação fosse dada pelo próprio usuário. Isso se tornou um problema muito sério devido à natureza móvel do celular, que não pode se apoiar no modelo de identificação da telefonia fixa (FCC, 2001).

Para solucionar tal entrave, os PDE's *(Position-Determining Equipmen*t) deveriam ter a capacidade de localizar o usuário a partir de uma ligação ao 911. Tais informações deveriam, ainda, ser enviadas à central de chamadas (*Call Center*) para que as devidas providências, no que concerne ao socorro ao usuário, pudessem ser tomadas em tempo hábil. Para garantir a eficiência do serviço de emergência, algumas regras foram estabelecidas e deveriam ser cumpridas por parte das operadoras de telefonia americanas.

As regras a serem obedecidas dividiram-se em duas fases. Na primeira fase, as operadoras deveriam identificar a célula e o número (ANI – *Automatic Number Information*) de onde a transmissão do usuário estivesse acontecendo. Como a célula pode cobrir uma área bem extensa, essa identificação fica longe do ideal. Essa fase foi cumprida no dia 01 de abril de 1998. Já na segunda fase, exigiu-se uma maior precisão na localização das unidades móveis, as operadoras necessitavam identificar o usuário dentro de uma precisão de 125m em pelo menos 95% das ligações.

A imposição feita pelo governo americano às empresas de telefonia fez com que fosse criada toda a infra-estrutura necessária para a localização de um usuário na rede, e com isso propiciou a criação de um novo mercado a ser explorado por essas operadoras.

2.4.2 Serviços de Informação

Os serviços de informação oferecem uma central de utilidade pública oferecendo serviços como páginas amarelas, acontecimentos, eventos e atrações. Assim, o usuário poderá ficar sabendo, por exemplo, onde fica o restaurante de comida japonesa mais próximo ou se existe algum evento cultural próximo ao hotel onde ele está hospedado.

Um exemplo dessa classe de aplicações são os guias de turismo eletrônicos, dentre eles destaca-se pelo pioneirismo o projeto GUIDE (DAVIES *et al*., 1999), desenvolvido pela Universidade de Lancaster. O projeto GUIDE é um guia turístico eletrônico para a cidade de Lancaster voltado para a utilização em dispositivos portáteis. Em outras palavras, ele possui o conhecimento da informação de localização física do usuário e fornece serviços específicos levando-se em consideração o seu interesse e o contexto atual. Por exemplo, se o usuário quer saber sobre a história da cidade, o GUIDE será capaz de permitir a construção de um roteiro pelos principais pontos históricos, dando a direção de como o turista poderá se locomover para a próxima atração. Ao avistar um monumento o usuário terá uma descrição completa do mesmo, podendo, através de uma rede sem fio, interagir com outros usuários e buscar informações mais completas na Internet.

Outros exemplos de guias digitais seriam o *Vindigo* (VINDIGO, 2002), *Citysync* (CITYSYNC, 2003) e o *Cyberguide* (ABOWD *et al*., 2000). Esses guias, além de possuírem as funcionalidades básicas de serviços de informação, também provêem um serviço de navegação que mostra em um mapa as direções passo a passo que devem ser seguidas para chegar a um determinado lugar.

No Brasil, a operadora de telefonia móvel Vivo lançou recentemente o seu primeiro serviço de localização, batizado de *Vivo Encontra*. Esse serviço utiliza a tecnologia de satélite *gpsOne* da *Qualcomm*, que capta o sinal de triangulação das estações radio-base

(ERBs) que compõem a rede CDMA 1x-RTT. A Vivo garante que o uso da solução híbrida gpsOne/ERBs elimina as falhas de localização e exatidão, oferecendo precisão de 5 a 50 metros . O serviço de localização da Vivo é composto por três aplicativos: o *Vivo Localiza*, o *Vivo Aqui Perto* e o *Vivo Onde Estou?* (VIVO, 2005).

O *Vivo Localiza* foi desenvolvido em parceria com a *Wiz Technologies* e é um serviço de localização de pessoas que identifica com precisão a posição de um usuário ou a localização de um ponto de interesse a partir do próprio celular. Disponível inicialmente nas cidades do Rio de Janeiro e de São Paulo, o sistema traz resultados com texto ou mapas e sugestões de rota, dependendo da funcionalidade.

O *Vivo Aqui Perto* é um aplicativo desenvolvido em parceria com a *Webraska* para procurar estabelecimentos comerciais, como bares, restaurantes e cinemas. A busca pode ser realizada por categoria, nome ou mesmo proximidade do usuário e além do endereço completo, o resultado traz mapa, com distância e sugestão de rota.

Por fim, *Vivo Onde Estou?* mostra a localização do celular, incluindo logradouro, número, bairro e cidade.

2.4.3 Serviços de Rastreamento

Consiste em monitorar o posicionamento de carros ou pessoas. São geralmente aplicados no rastreamento de carros roubados, na administração de frotas de caminhões, na distribuição de mercadorias e também servem para ajudar paramédicos e equipes de resgate a localizar pessoas em situações de emergência.

Como exemplo dessa classe de aplicações, temos o Autotrac, cuja tecnologia é utilizada para comunicação, transmissão e integração móvel de dados, monitoramento e rastreamento de frotas via satélite e soluções para gerenciamento logístico e de risco. O

sistema é adotado por empresas do setor de transportes de cargas nos modais rodoviário, ferroviário e hidroviário; serviços públicos, como as concessionárias para distribuição de energia elétrica; e órgãos do governo, como secretarias estaduais de Fazenda e de Segurança Pública (AUTOTRAC, 2005).

O sistema chamado Autotrac Tradicional permite o gerenciamento logístico e de risco e comunicação móvel de dados dos veículos equipados com a tecnologia composta por MCT (Terminal de Comunicação), OBC (Computador de Bordo Inteligente), atuadores (trava de baú, trava de tanque, imobilizador de freio, bloqueio de motor, etc.) e sensores (desengate, baú, carona, velocidade, violação do equipamento, pânico, etc.).

No quesito segurança, possibilita o gerenciamento de risco via satélite e das exceções, a prevenção e eliminação de riscos e programação de áreas de risco. Além disso, possibilita a percepção da emergência e atuação remota em tempo real, a localização exata do veículo e o envio e recebimento de mensagens em segundos. A comunicação bidirecional também é garantida entre os usuários dos veículos rastreados e suas bases de operações, colaborando para que os veículos equipados sejam monitorados constantemente e que seja feito o registro de todos os eventos da viagem. Dessa forma, é possível enviar mensagens de emergência quando necessárias e até realizar funções pré-programadas para as atividades logísticas e de gerenciamento de risco.

Em 2004, a Autotrac lançou comercialmente o *Autotrac Caminhoneiro* especialmente para os caminhoneiros autônomos, que passam a integrar um dos principais públicos-alvo da empresa. O *Autotrac Celular* está em fase final de desenvolvimento e o lançamento está previsto para o segundo semestre de 2005. O produto terá as mesmas funcionalidades do *Autotrac Tradicional* e do *Autotrac Caminhoneiro*. O que muda é o tipo de tecnologia usada para comunicação e a sua cobertura. No lugar de satélite, o novo produto usará o celular. Esse tipo de serviço está sendo desenvolvido para atender empresas de

transporte com atuação em perímetro urbano. É o caso de empresas de entregas, transportadoras e atacadistas com atuação em áreas urbanas, polícias, concessionárias de energia e frotas de rádio táxi.

2.4.4 Navegação

Dá informações ao usuário de como ele pode se movimentar do local onde se encontra até um ponto específico. Os sistemas de navegação podem determinar rotas ideais levando-se em consideração tanto a distância mínima quanto o tempo mínimo gasto na movimentação. Por exemplo, uma pessoa quer se deslocar de sua casa para a casa de um amigo. Primeiramente, ela consulta os caminhos existentes entre os dois locais, depois pode obter qual o menor trajeto a ser percorrido e, por fim, baseado nas condições de tráfego, saber por qual caminho a pessoa deve seguir para chegar no tempo mínimo.

Um exemplo de sistema de turismo que também se enquadra na classe em questão é o sistema *Trekker* (VISUAIDE, 2003), que consiste em uma aplicação para dispositivos do tipo PocketPC, adaptados para pessoas cegas. Ele oferece a possibilidade de as pessoas com deficiência visual identificar sua localização, criar rotas e receber, através de comandos de voz, informações de navegação para chegar a um ponto de interesse determinado.

Desenvolvido pela empresa francesa *Webraska Mobile Technologies*, o Apontador Duo (WEBRASKA, 2005) é um guia de endereços portátil que permite visualizar ruas, mapas, traçar rotas, gravar endereços e percursos mais utilizados em um Pocket PC. Com uma grande cobertura em cidades do Brasil, o Apontador Duo acoplado a um receptor GPS é uma ferramenta extremamente prática para obter a melhor rota entre dois pontos e que orienta através de mapas e comandos de voz até o destino desejado.

Funciona em dois modos (DUO – com ou sem conexão à Internet), permitindo que o usuário escolha a melhor forma de trabalho, de acordo com a sua necessidade no momento.

No modo *off-line*, o usuário poderá utilizar-se de mapas previamente instalados em seu dispositivo para permitir o recálculo automático de rota e visualizar mapas detalhados das ruas e avenidas. Já no modo *on-line*, pode-se obter o acesso aos servidores remotos através da Internet, permitindo o cálculo das rotas desviando dos pontos de congestionamento em tempo real, fornece mapas com a informação do trânsito em tempo real e o cálculo de rotas entre duas cidades (para o centro de qualquer município do Brasil). Para soluções corporativas, os mapas a serem utilizados podem ser personalizados, como, por exemplo, inserindo pontos de interesse do cliente nos mesmos.

3 TRABALHOS CORRELATOS

Estudos em diversas áreas, como banco de dados, tecnologia de posicionamento, engenharia de software, entre outras, tem sido elaborados para tentar facilitar a construção de sistema LBS de forma a aproveitar todo o leque de novos serviços que podem ser oferecidos através dessas aplicações. Uma série de artigos propõem arquiteturas de referência e aplicações LBS (CHEN *et al*., 2004) (NORD *et al*., 2002) (RUI, *et al*., 2001), outra linha de pesquisas propõe o desenvolvimento de artefatos de software que facilitem a construção dos sistemas baseados em localização (SATOH, 2004) (JÄRVENSIVU *et al*., 2004) (TSELIKAS *et al*., 2004). Neste capítulo descreveremos alguns trabalhos relevantes, tendo como enfoque o estudo das arquiteturas e *frameworks* propostos na literatura.

3.1 Arquiteturas

A iniciativa OpenLS (OGC, 2004) é voltada para o desenvolvimento de especificações de interfaces que facilitem o uso da localização e de outras formas de informação espacial sobre o ambiente da Internet sem fio. O objetivo dessa iniciativa é especificar um conjunto de interfaces padrões e protocolos nos quais desenvolvedores possam usar para integrar dados geoespaciais e recursos de geoprocessamento dentro de serviços de localização e da infra-estrutura de telecomunicação, disponibilizando essas capacidades para uma grande variedade de aplicações.

A especificação OpenLS foi aprovada pelo consórcio OpenGIS[5] (OGC) em janeiro de 2004. A plataforma resultante, chamada de *GeoMobility Server*, fornece funções básicas para que as aplicações baseadas em localização sejam construídas. Ela usa interfaces

[5] O OGC (*Open Geospatial Consortium*) 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. Os produtos do trabalho do OGC são apresentados sob forma de especificações de interfaces e padrões de intercâmbio de dados.

para permitir o acesso aos mecanismos de localização da rede e a um conjunto de serviços denominado *OpenLS Core Services.* Os serviços especificados encontram-se descritos a seguir:

a) Serviços de Geocodificação/Geocodificação Reversa (*Location Utilities Service*): o primeiro converte a descrição simbólica de um local em uma descrição física ou geográfica do mesmo, isso em termos de suas coordenadas. Por outro lado, o último faz exatamente o contrário, mapeando uma posição física em um endereço simbólico;
b) Serviço de Diretório (*Directory Service*): fornece a capacidade de pesquisar por um ou mais pontos de interesse (um lugar, produto ou serviços com posições fixas) ou áreas de interesse (um bairro delimitado por um polígono, uma quadra). Um ponto de interesse (POI) é definido como um tipo abstrato de dados que representa um local ou uma entidade que possui uma posição fixa, podendo ser usada como um ponto de referência ou um marco em um sistema de localização. Um POI deve conter um nome, um tipo, uma categoria, um endereço e um conjunto de outras informações que descrevam o lugar, o produto e/ou serviços relacionados a ele. Já a área de interesse (AOI) é um tipo abstrato de dados que define uma região de interesse do usuário, sendo delimitada por um círculo ou um polígono. O AOI é utilizado como parâmetro para buscas ou para a visualização de uma área determinada;
c) Serviço de Apresentação (*Presentation Service*): permite a visualização da informação espacial através de mapas, imagens, rotas e em textos. É usado para apresentar mapas destacando rotas entre dois pontos, pontos de interesse, áreas de interesse, localizações e/ou endereços;

d) Serviço de Determinação de Rotas (*Route Determination Service*): disponibiliza a capacidade de encontrar o melhor caminho entre dois pontos que satisfaçam às necessidades do usuário.

A Figura 3.1 mostra como o *GeoMobility* Server (GMS) se relaciona com outros componentes de uma arquitetura LBS. O Portal deve possuir funções para suportar o gerenciamento de sessões, autenticações e manipulação de requisições de usuários. Também poderá dar suporte a funções de tarifação, administração de privacidade, bem como funcionar como um *proxy*, dando suporte para a personalização de conteúdo. O Portal pode, ainda, funcionar apenas como uma interface com sistemas externos à aplicação LBS transferindo solicitações para essas aplicações.

O GMS pode conter aplicações locais que, através de uma API, acessam o *OpenLS Core Services* que, por sua vez, utilizam interfaces OGC para obter o conteúdo da localização necessário para realizar suas funções. Tais conteúdos seriam: dados de mapas, redes de rotas, endereços, informações de navegação, diretórios com informações de lugares, produtos ou serviços. O GMS também possui a capacidade de acessar outros GMSs e bases de dados localizadas em outros servidores.

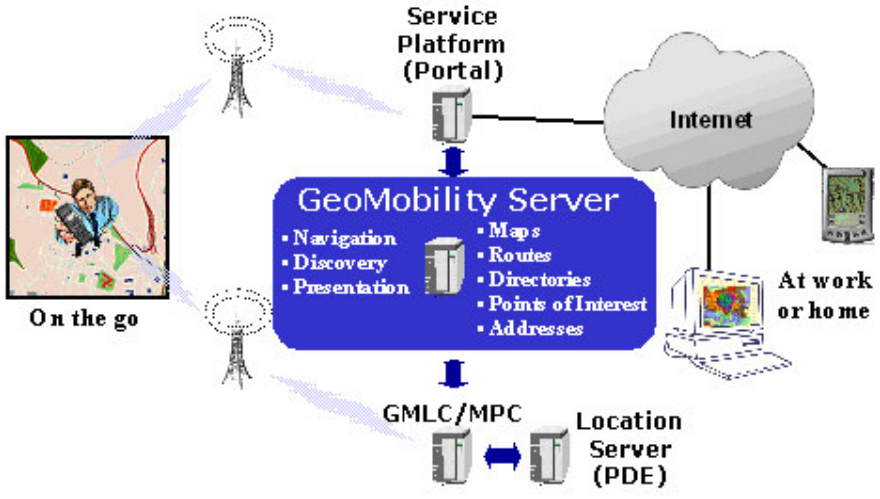

Figura 3.1 Arquitetura Geral do OpenLS (OGC, 2004)

Uma outra arquitetura para serviços de localização é proposta por Leonhardi e Rothermel (2002). Neste trabalho, é apresentado um modelo do serviço de localização (ou API genérica), definindo a semântica das consultas de posição (*position*), de áreas (*range*) e de proximidade (*nearest neighbor*). Para suportar tal modelo, foi definida uma arquitetura distribuída e hierárquica dos servidores, mostrada na Figura 3.2. Seus componentes e suas funcionalidades são descritas a seguir:

a) clientes e sistemas sensores: os clientes do serviço de localização são aplicações sensíveis à localização do usuário, que podem estar executando em um dispositivo móvel ou em uma estação de trabalho e acessam o servidor através de uma API. O cliente de um dispositivo móvel também se encontra equipado com um sensor de posicionamento ou está sendo rastreado por um. Um sistema sensor determina a localização de um ou mais objetos, os quais estão associados com um servidor de localização;

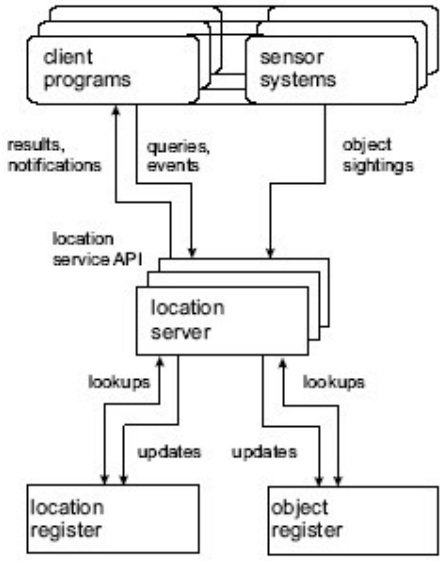

Figura 3.2 Principais componentes da arquitetura (LEONHARDI; ROTHERMEL, 2002)

b) Servidor de Localização (*Location Server*): cada servidor de localização é responsável por uma determinada área geográfica e por armazenar a localização de todos os objetos móveis que estão atualmente dentro dessa área. Se um objeto sair da região, então é feita uma transferência das informações para o servidor responsável pela área para o qual o objeto se deslocou. Esse servidor também é responsável por responder as consultas dos usuários a respeito da localização de outros objetos móveis. Para realizar tais tarefas o servidor acessa dois registros: *object register* e *location register*;

c) *object register*: guarda informações gerais sobre os objetos móveis ou informações que são determinadas pelo serviço de localização, como a velocidade máxima de um objeto. Neste registro, também é armazenado o endereço do servidor responsável pela área onde o objeto está localizado;

d) *location register*: armazena a associação entre os servidores de localização e as áreas para o qual eles serão responsáveis.

Na Figura 3.3 pode-se observar uma arquitetura para a construção de sistemas sensíveis à localização, definida por Agre *et al*. (2002). Concentra-se na definição do *Location Service Module* (LSM) uma camada de *midleware* que simplifica o desenvolvimento dos serviços baseados na localização, já que oferece uma interface uniforme que encapsula detalhes relacionados aos métodos de localização e fornece diversas funções úteis, tais como: comutação de múltiplas tecnologias, estimativas de erros, rastreamento, combinação de tecnologias de localização e de determinação cooperativa da localização.

O LSM é dividido em uma camada de adaptação da localização (LAL - *Location Adaptation Layer*) e em uma camada dependente da localização (LDL - *Location Dependent Layer*). A razão para a separação em duas camadas é desvincular a aplicação dos avanços e das mudanças nas diferentes tecnologias de posicionamento, disponibilizando uma

informação de localização padrão, a fim facilitar o desenvolvimento das mais variadas aplicações.

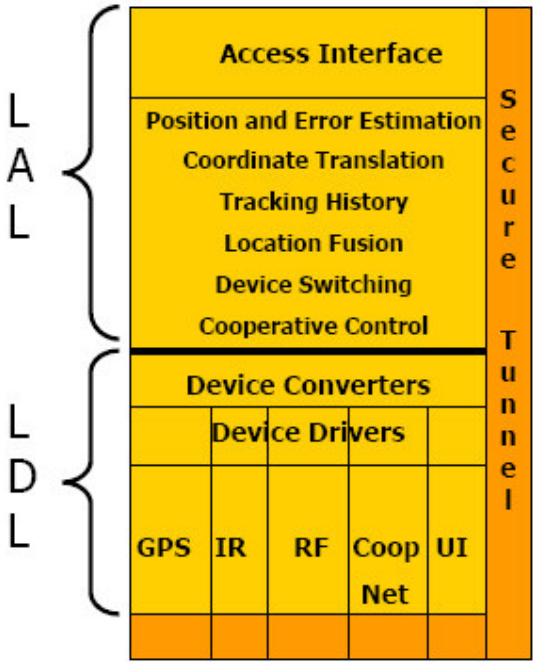

Figura 3.3 Arquitetura do LSM (AGRE *et al*, 2002)

O LAL controla os cálculos para a determinação da posição, sendo dependente da aplicação, no sentido que é configurável para uma dada aplicação, embora baseado em um conjunto comum de funções. As funções primordiais do LAL são: a entrega dos dados de localização às camadas superiores em formato padrão através de uma interface de acesso; o cálculo e tradução das coordenadas; a comutação transparente entre tecnologias de localização; a fusão dos dados de posição das múltiplas fontes; o cálculo de estimativas de erro da posição; o rastreamento e integração de dados da posição para o incremento da precisão e, por fim, o controle de métodos distribuídos para estimar a posição.

Por sua vez, o LDL é dependente dos métodos de determinação da posição que serão empregados. Fornece uma abstração genérica ao LAL através das rotinas de conversão

que escondem detalhes dos *drivers* para os múltiplos dispositivos, como GPS, sinais infravermelhos (IR), radiofreqüência (RF) ou inserção de dados pelo usuário (UI).

Além das APIs anteriormente descritas, o acesso direto aos dispositivos pode ser obtido através de uma facilidade denominada *secure tunnel*. Ele reconhece as diversas propriedades e características próprias de cada dispositivo. Por exemplo, os *waypoints*[6] são suportados freqüentemente por dispositivos de GPS, mas não suportados atualmente pelo LSM. O LSM não pretende inibir o uso de características especiais, mas suportar as características mais comuns.

3.2 Frameworks

Nesta seção será feita uma análise dos *frameworks*[7] relevantes encontrados na literatura para a construção de aplicações LBS, ressaltando seu funcionamento, suas qualidades e deficiências.

Em (LIU, 2002) foi desenvolvido um *framework* para aplicações LBS, que tem como objetivo facilitar a construção de sistemas baseados em localização a partir da reutilização das camadas referentes à comunicação sem fio, dando suporte às mais populares tecnologias disponíveis no mercado. O *wireless framework* fornece um conjunto de classes responsáveis por encapsular detalhes de implementação relacionados à comunicação de rede sem fio, mais especificamente ao tipo de comunicação a ser utilizada (CDPD, GSM, RFM96), aos protocolos (TCP/IP, UDP/IP, RADIO/ACK, RADIO/UNACK), tipo de criptografia e

[6] São pontos geodésicos específicos gravados na memória do receptor GPS. Pode ser visto como uma coordenada digna de menção, que contém algum evento de interesse. Geralmente representam lugares específicos, como cidades, praças, pontes, cruzamentos, etc. Um waypoint é composto pelos seguintes campos: latitude, longitude, altitude, nome, comentários, data de criação.

[7] É um modelo reutilizável representado por um conjunto de classes que colaboram entre si e constituem a arquitetura principal de uma aplicação ou subsistema (MATTSSON, 1996). A idéia principal de um *framework* é permitir que um conjunto de recursos comuns seja reaproveitado para cada novo software criado. Sua utilização facilita o desenvolvimento de aplicações ao permitir que o programador trabalhe com menos elementos, abstraindo os detalhes de implementação. Outro benefício é a modularidade. (FAYAD et al.,1999).

compressão das informações enviadas ou recebidas. Omitindo dos desenvolvedores, por exemplo, informações de baixo nível referentes ao envio e recebimento de mensagens para uma tecnologia de comunicação específica. Esse *framework* encontra-se dividido em quatro classes principais: modem_type, protocol_selection, encryption_selection e compression_selection. Todas elas derivadas da super classes wireless_communication, como mostra a Figura 3.4.

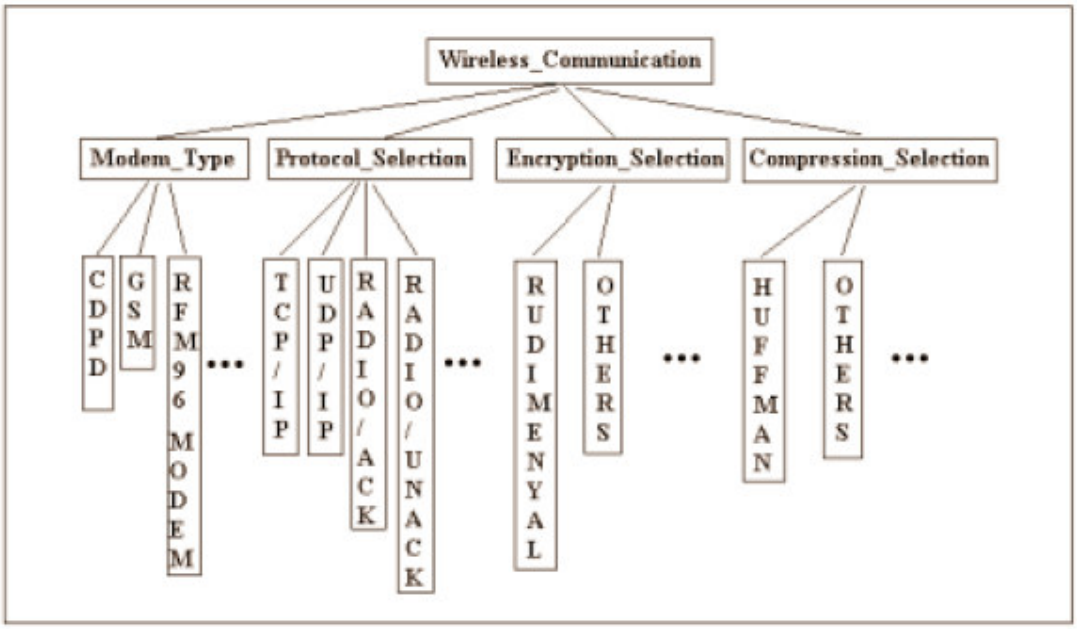

Figura 3.4 Hierarquia de classes do *wireless framework*

O *framework* proposto está baseado em uma arquitetura aberta e foi desenvolvido segundo o paradigma da Orientação Objetos permitindo, assim, uma incorporação fácil e amigável de novas tecnologias de rede à sua infra-estrutura. A implementação em Java dá aos *softwares* desenvolvidos uma independência com relação à plataforma utilizada. No entanto, este trabalho está restrito à reutilização das funcionalidades de comunicação de redes sem fio.

Outro trabalho relacionado refere-se ao LBSFramework (ANDERSEN *et al*., 2003) que possui como objetivo prover mecanismos que possibilitem a utilização de duas bases de dados distintas como se fosse uma única fonte de informação, baseando-se na existência de dois conjuntos de dados: os dados geográficos, que servem para prover as

informações baseadas na localização e os dados de conteúdo, que relacionam as informações de localização com informações contextualizadas que possam vir a ser úteis para o usuário (pontos de interesse, situação do trânsito, limite de velocidade). A Figura 3.5 mostra o *framework* abstrato proposto. Este divide os componentes em cinco camadas lógicas (*Data Layer*, *Application Layer*, *Service Layer*, *Client Layer* e *Position Layer*).

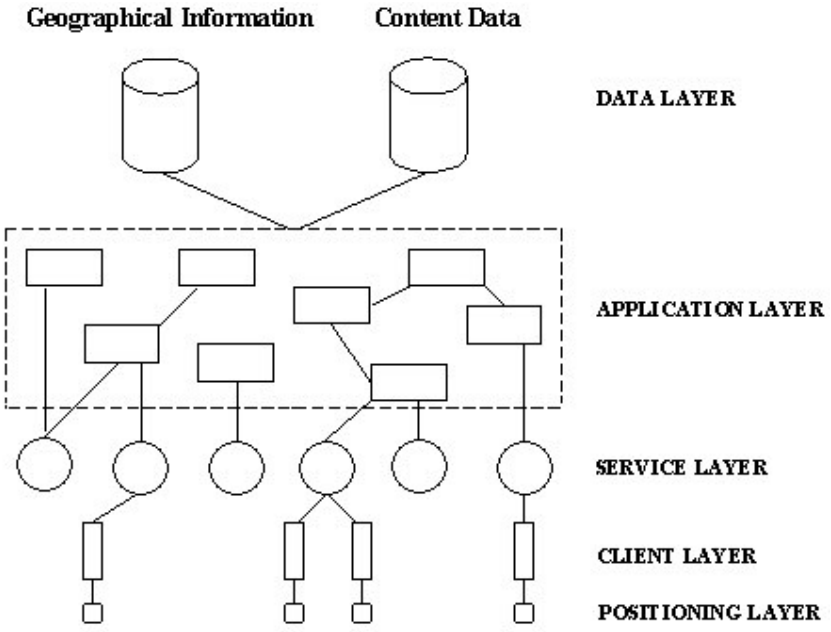

Figura 3.5 Arquitetura do *framework* abstrato (ANDERSEN *et al*, 2003)

A *Position Layer* tem com objetivo obter a posição dos usuários móveis, sendo formada pelas tecnologias de localização. A camada cliente representa as aplicações disponibilizadas no dispositivo móvel para a comunicação com os equipamentos da *Position Layer* e para acessar os serviços (interfaces) oferecidos pela *Service Layer,* como serviços de *sms*, *web*, *e-mail*, etc. A *Application Layer* é responsável por possibilitar a comunicação da camada de serviços com a camada de dados.

Para validar o *framework*, foi implementado um protótipo LBS que fornece basicamente três serviços: “Procurar pontos de interesse próximos (POI)”, “Obter a rota para chegar aos POI’s” e “Obter um mapa para o POI”.

Neste trabalho não existe qualquer forma de gerenciamento da informação de localização dos dispositivos móveis, sendo que a posição dos usuários só é fornecida no ato da solicitação dos serviços ao servidor. Tal abordagem inviabiliza a implementação de serviços proativos baseados na ocorrência de eventos ou condições satisfeitas (serviços *push*), impossibilita também o rastreamento da localização dos usuários (frotas de caminhões) e não permite a realização de consultas temporais, uma vez que a localização do usuário não é armazenada.

O Universal Location Framework (LARA, 2003) caracteriza-se como um conjunto de componentes de *software* que possibilitam agregar múltiplas tecnologias de localização, permitindo a transição entre elas. A Intel tem visto o Universal Location Framework (ULF) como um bloco que pode habilitar a criação de aplicações e serviços inovadores e vantajosos.

O ULF tem como proposta não restringir o uso de qualquer tecnologia de localização específica. Para isso, usa uma camada que permite a integração (fusão) dessas múltiplas tecnologias. Atualmente, os desenvolvedores devem escrever aplicações em cima de interfaces de baixo nível específicas da tecnologia. Por exemplo, para escrever aplicações que usam GPS, o desenvolvedor deve usar uma API para a porta de comunicação serial para ler os dados. ULF apresenta uma API simples para a aplicação que permite tomar vantagem de todos os métodos de localização de forma transparente.

A Figura 3.6 mostra um diagrama de blocos representando a arquitetura ULF. Ela apresenta-se dividida em três camadas: *sensors*, *measurements* e *fusion*. A camada de sensores consiste nos dispositivos de *hardware* sensíveis à localização do usuário e nos

drivers para capturar a informação de localização bruta, isto é nativa do dispositivo. Atualmente, existe suporte para duas tecnologias: GPS e WLAN.

A camada de conversão é formada por um conjunto de serviços, um para cada tipo de tecnologia de localização suportado, que incorporam algoritmos para converter os dados brutos em informações de localização descritas em um formato padronizado (posição, incerteza, tempo). A principal função da camada de fusão é fazer continuamente a junção do fluxo de medidas recebidas dos serviços de conversão para uma simples estimativa de posição. A estimativa resultante é mais precisa que uma medida levando-se em consideração uma tecnologia de localização individual.

Essa proposta restringe-se à questão da obtenção da localização do cliente móvel sem se preocupar com os protocolos de atualização dessa informação e com o seu gerenciamento.

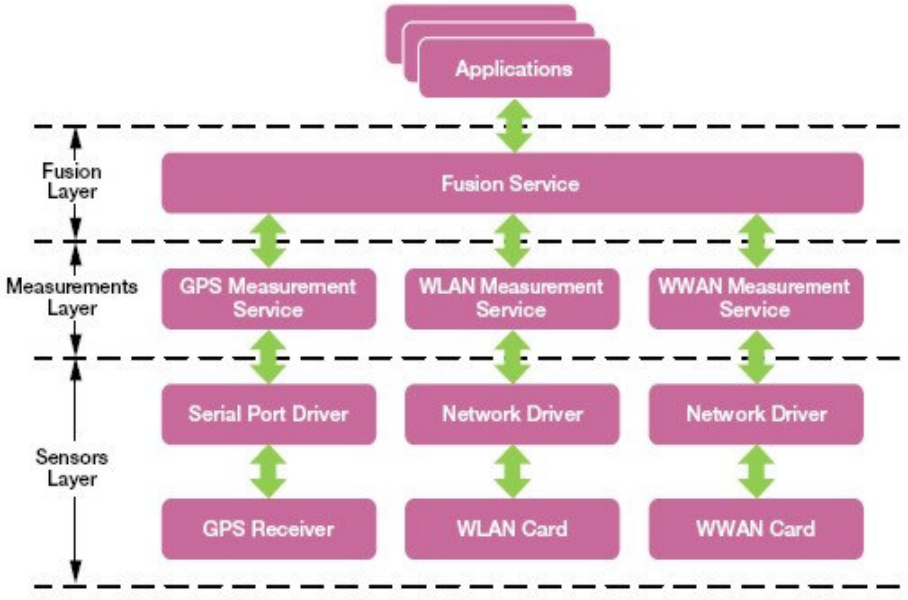

Figura 3.6 Arquitetura do UFL (LARA, 2003)

Stojanovic (2001) apresenta o desenvolvimento do *framework* iSTOMM, que fornece uma variedade de serviços espaço-temporais obtidas de diferentes fontes de informação através da estrutura da Internet e da posição dos terminais móveis. Também provê

suporte para a apresentação/visualização dos resultados a serem mostrados em um cliente móvel ou em um terminal Web estático em forma de mapas, relatórios e mensagens.

Observa-se, portanto, que nenhum dos trabalhos encontrados na literatura se propõe ao desenvolvimento de um *framework* específico para gerenciamento da informação de localização, que permita uma completa reutilização dessas funcionalidades, que seja independente do sistema gerenciador de banco de dado espacial e facilmente acoplável a uma aplicação LBS. No entanto, evidencia-se que os trabalhos aqui citados podem ser perfeitamente utilizados em conjunto com o FRAGIL, uma vez que abordam características e áreas complementares relacionadas aos sistemas baseados em localização.

4 FRAGIL: FRAMEWORK PARA O GERENCIAMENTO DA INFORMAÇÃO DE LOCALIZAÇÃO

Neste capítulo será feito um estudo detalhado a respeito da concepção, da modelagem e do desenvolvimento do *framework* proposto, isto é, apresentaremos sua arquitetura, exemplificaremos seu funcionamento no contexto de uma aplicação LBS através de um estudo de caso e descreveremos seus componentes internos. Em seguida, apresentaremos questões relacionadas à modelagem e implementação do FRAGIL.

4.1 Arquitetura do Framework

Nesta seção, será apresentada a arquitetura proposta para um *FRamework* de Gerenciamento da Informação de Localização, FRAGIL que busca possibilitar a reutilização de funcionalidades de gerenciamento da informação de localização oferecendo diversos protocolos para a atualização dessa informação, permitindo a obtenção da mesma através do dispositivo móvel ou de dispositivos da rede e oferecendo diversas consultas espaciais úteis em aplicações LBS.

Nesta arquitetura, ilustrada na Figura 4.1, podemos observar que o *framework* é organizado segundo um modelo multicamadas, englobando três camadas: *Location Server*, *ClientAPI* e *ApplicationAPI* . O *Location Server* (*LS*) é responsável por gerenciar a informação de localização dos objetos móveis (*mo*), oferecendo para a aplicação LBS e seus clientes serviços relacionados à posição de tais objetos. Para isso, o *LS* é composto por vários componentes responsáveis pela execução de tarefas específicas, tais como: definição e controle dos protocolos de atualização da informação de localização dos *mo* (*UpdatePosService*), cadastro de um novo *mo* a ser monitorado (*RegisterService*), elaboração

de consultas (*QueryService*), controle de eventos relacionados ao posicionamento de um ou mais objetos (*EventService*) e manutenção da informação de localização.

A camada *ClientAPI* é uma interface de comunicação do cliente móvel com o *LS*, essa camada permite solicitação de registro do *mo* e a atualização de sua localização, de acordo com o protocolo de atualização escolhido. Por outro lado, a *ApplicationAPI* é a camada responsável pela comunicação do *LS* com o servidor que contém a semântica da aplicação LBS, ao qual denominamos *LBS Server* (*LBSS*).

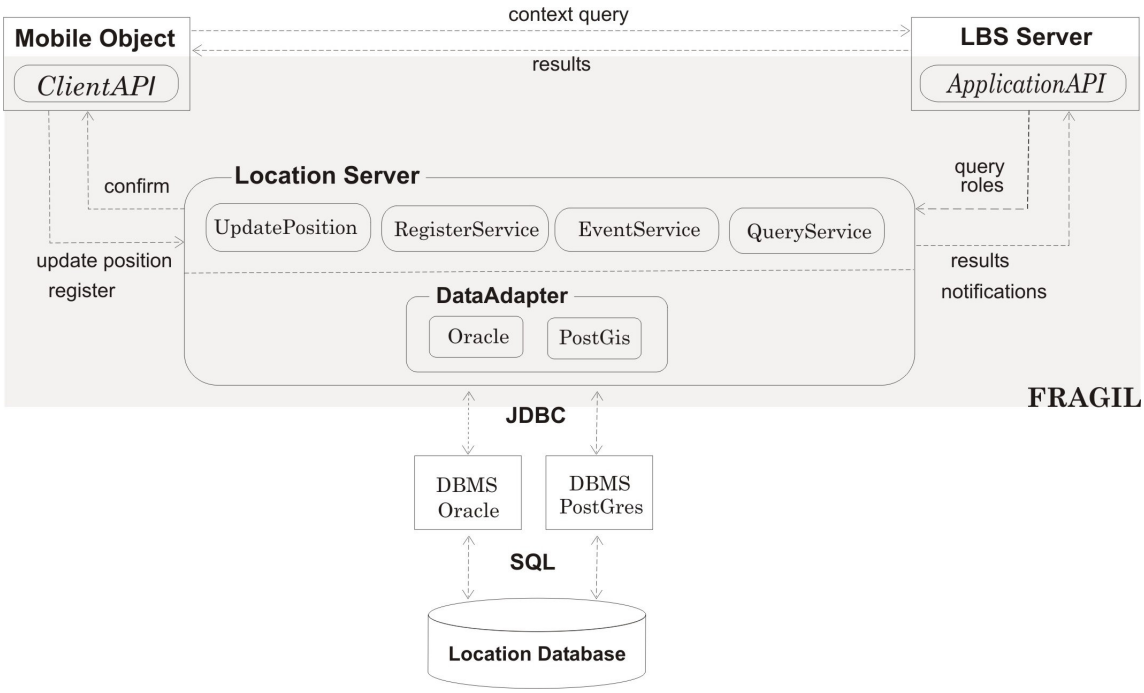

Figura 4.1 Arquitetura geral do FRAGIL

Em um cenário típico, teremos um certo número de *mos* que a qualquer momento poderão fazer consultas contextualizadas ao *LBSS* utilizando a semântica da aplicação, um exemplo de consulta poderia ser: “quais os restaurantes chineses mais próximos da minha posição?”. O *LBSS* possui informações sobre os dados geográficos que servem para prover as informações baseadas na localização e os dados de conteúdo que relacionam as informações

de localização com informações contextualizadas que possam vir a ser úteis para o usuário (pontos de interesse, situação do trânsito, limite de velocidade, etc). Utilizando o exemplo acima, o *LBSS* teria uma lista de restaurantes, suas especialidades e sua localização geográfica. Porém, não possuiria qualquer informação sobre a atual localização dos *mos*. Tais informações são administradas pelo *Location Server* (*LS*) que, baseado em um protocolo de atualização e de gerenciamento de privacidade, constantemente recebe mensagens informando a posição atual de cada *mo*. Assim, para responder uma consulta de um cliente o *LBSS* deverá utilizar-se de uma API específica (*ApplicationAPI)* para obter as informações de localização a partir do *LS*.

Podemos verificar como o FRAGIL funciona no contexto de uma aplicação LBS, tomando como exemplo o seguinte cenário: consideremos um sistema que propicie suporte à equipes móveis de emergência, permitindo a gestão dessas equipes através do acompanhamento em tempo real de seu posicionamento geográfico bem como a disponibilização de informações georeferenciadas que facilitem a execução das tarefas a serem executadas pelas mesmas. Nesse cenário, os *mos* a serem monitorados seriam ambulâncias. A partir de uma central de controle, são recebidos os chamados solicitando ajuda médica de emergência em qualquer local da cidade, assim uma informação fundamental para este sistema seria responder, por exemplo, à seguinte questão: "Quais ambulâncias estão mais próximas do local de um determinado acidente?". A seguir, ilustraremos como essa solicitação seria tratada por cada componente do *framework*:

- Primeiramente, cada *mo* deve, através da *ClientAPI,* registrar-se no *LS*. A partir desse momento o *mo* passa a fornecer ao *LS* informações a respeito da sua posição geográfica;
- Através de uma aplicação, o operador do sistema entra com a solicitação: "Quais ambulâncias estão mais próximas do acidente ocorrido no endereço X"? Essa

solicitação é, de início, processada pelo *LBSS* que contém informações contextualizadas para essa aplicação;

- O *LBSS* utiliza-se de um serviço de geocodificação para traduzir o endereço X do acidente em coordenadas geográficas. Porém, não possui qualquer informação a respeito da localização de suas equipes de emergência;
- O *LBSS*, utilizando-se da *ApplicationAPI*, solicita ao *LS* a lista de *mos* que estão mais próximos das coordenadas do local do acidente;
- No *LS,* a requisição é recebida e processada por um componente específico, no caso o *QueryService*. Este, por sua vez, envia uma consulta parametrizada ao *DataAdapter* que extrai a informação necessária do banco de dados de localização retornando-a ao *QueryService*;
- Por fim, o *QueryService* prepara a lista de ambulâncias e a envia para o *LBSS*, que exibe a informação para o usuário final.

Para manter o *framework* independente de SGBD e das linguagens de consulta espacial específicas de cada arquitetura proprietária, foi criado um componente, inspirado na arquitetura do *framework* IGis (MIRANDA *et al*., 2002), denominado *DataAdapter*, responsável pela tradução de consultas realizadas no padrão OpenGIS para a linguagem específica de cada sistema de banco de dados. Inicialmente estaremos disponibilizando a interação com os módulos espaciais do Oracle.

O *LS* é composto pelos seguintes componentes: o *UpdatePosService*, que permite a atualização da posição atual dos objetos móveis; o *RegisterService*, que administra o cadastro de um novo objeto a ser monitorado e é responsável pela definição das políticas de atualização a serem utilizadas de forma a otimizar o acesso às informações de localização; o *QueryService*, que fornece uma série de consultas baseadas na posição dos *mo* e o

EventService, que permite o cadastro e o controle dos eventos relacionados ao posicionamento de um ou mais objetos.

Nas subseções seguintes descreveremos com detalhes os componentes internos do *LS*.

4.1.1 *RegisterService* - Componente de registro do objeto móvel

Para que a posição de um objeto possa ser gerenciada é necessário que este seja registrado junto ao módulo de gerenciamento da informação de localização. Para tanto, algumas informações iniciais deverão ser fornecidas, tais como: posição inicial, tempo inicial e tipo de política de atualização a ser adotada.

Cada objeto móvel que deseja ser monitorado deverá, através da *ClientAPI*, enviar uma mensagem registrando-se no *Location Server*. Este, por sua vez, cria uma sessão de monitoramento para o objeto e o envia uma mensagem de confirmação do serviço. A iniciativa de solicitar o registro de um objeto também poderá ser feita por um servidor nos casos em que o objeto não tem a capacidade de obter o seu posicionamento por si só, dependendo de um sistema de posicionamento baseado na rede. A partir desse momento, um identificador será atribuído ao objeto, permitindo que seja referenciado tanto pelos módulos da aplicação quanto pelo componente de gerenciamento de localização.

Dependendo do tipo de aplicação que o usuário estiver interessado em desenvolver, poderá haver duas formas de armazenar a posição de um determinado objeto. A primeira delas, seria ter para cada objeto uma única entrada no banco de dados que representaria a provável posição atual daquele objeto específico. De outra forma, poderíamos ter uma segunda abordagem onde para cada *mo* existe um conjunto de entradas no banco de dados, definindo sua posição no tempo *t*. Essa última abordagem é essencial para aplicações

que necessitam obter a localização dos usuários tanto no presente quanto no passado, um exemplo seriam as aplicações de rastreamento. Tal abordagem é a que atualmente encontra-se disponibilizada no *framework.*

4.1.2 *UpdatePosService* - Componente de atualização da posição

Para controlar a transmissão da informação de localização, existem diferentes protocolos de atualização que podem ser adotados. Tais protocolos usam várias propriedades especiais da informação de localização para transmití-las da forma mais eficiente possível, objetivando minimizar o número de mensagens para atualizar o servidor, sem perder, contudo, o efetivo grau de precisão desejado. A localização do dispositivo móvel é determinada diretamente por um sistema de posicionamento acoplado ao próprio dispositivo ou está armazenada em outro servidor de localização.

Segundo Leonhardi e Rothermel (2000), os protocolos de atualização são classificados em três classes: *querying*, *reporting* e *combined protocols*, onde cada classe possui um número típico de variantes. Cada um desses protocolos possui propriedades características e que se modificam para um dado ambiente ou para certos requisitos da aplicação.

Um protocolo é classificado como *querying* se o servidor decide quando solicitar a informação de localização do *mo*. Nesse caso, o *mo* pode ser implementado de forma simples, uma vez que não necessita analisar extensas informações sobre o seu estado ou realizar processamentos lógicos complicados. Isso pode ser importante para dispositivos de pequeno porte.

Na forma mais simples, denominada *simple querying*, o servidor solicita a informação de localização do *mo* sempre que a aplicação necessitar de tal informação e,

portanto não necessitaria ficar armazenada no servidor. Isso implica em um alto grau de precisão da informação de localização, mas também em um grande número de troca de mensagens, se a mesma for requerida de forma freqüente. O tempo de resposta do servidor é também comparativamente alto, uma vez que o mesmo deverá contactar o cliente a cada nova solicitação. Por outro lado, esse protocolo deve ser utilizado se as configurações de privacidade do usuário não permitirem que sua localização seja armazenada em qualquer servidor de localização a não ser no seu próprio dispositivo.

Uma otimização do protocolo anterior, em que o servidor armazena uma cópia da última atualização transmitida, é conhecida como *cached querying*. Nesse caso, quando a informação de localização de um objeto é solicitada, o servidor analisa se a informação que possui é satisfatória para a aplicação e a retorna, caso contrário comporta-se como no *simple querying* e solicita a informação ao *mo*. Por fim, está disponível a técnica em que, a cada intervalo de tempo determinado *t*, o servidor solicita a localização ao *mo*, a esse protocolo denominamos *periodic querying*.

Outra classe de protocolos que merece destaque são os *reporting*. Esta é a classe de protocolos atualmente suportada pelo *framework*. No caso desses protocolos, a iniciativa da atualização parte do *mo*. Este sabe qual foi a última informação enviada para armazenamento no servidor e, portanto conhece a informação armazenada nele.

Uma mensagem de atualização é enviada sempre que uma comparação com a posição atual atinge algum parâmetro de configuração predeterminado como, por exemplo, tempo e distância. O servidor pode assim calcular o máximo de incerteza do valor armazenado a partir do valor limite parametrizado. Com esse protocolo, o servidor sempre responde às solicitações das aplicações consultando diretamente o valor atualizado em sua base. Dessa forma, ele somente pode retornar a informação de localização com a precisão determinada pelo valor limite parametrizado, ainda que a aplicação solicite uma informação

mais precisa. O tempo de resposta do servidor é relativamente curto, já que o mesmo não necessita entrar em contato com a fonte. Tal protocolo é usualmente mais eficiente que a classe de protocolos anteriormente citados se a informação de localização é requerida freqüentemente ou se o servidor administra a ocorrência de eventos baseados na posição de um ou mais objetos.

Na versão denominada *simple reporting*, o *mo* atualiza a localização no servidor toda vez que o sistema de posicionamento determinar uma nova localização para o objeto móvel. O número de mensagens enviadas depende da quantidade de atualizações feitas pelo sensor que está sendo utilizado, podendo chegar a níveis elevados.

Já no *time-based reporting*, a informação é transmitida periodicamente, depois de decorrido certo intervalo de tempo *t*. A taxa de atualização é fixa e não depende do comportamento do objeto móvel, o qual garante uma certeza temporal, mas não espacial da informação. Se o objeto move-se de forma lenta ou não se move, então uma diferença mínima ou inexistente é observada entre a informação real e a informação contida no servidor. Por outro lado, se o objeto está se movendo de forma rápida, dependendo do intervalo de tempo utilizado, o número de mensagens enviadas pode não ser suficiente para garantir uma elevada precisão da posição.

Em outro tipo de protocolo, classificada como *distance-based reporting*, a fonte envia uma mensagem de atualização toda vez que uma distância geográfica entre a posição atual e a última posição enviada para o servidor superar um valor limite *d*. Como esse protocolo envia um número maior de mensagens se o objeto estiver se locomovendo rapidamente e um número menor de mensagens se o mesmo estiver se deslocando de forma lenta ou estiver parado, ele é mais eficiente para objetos que realizam movimentos esporádicos entre períodos de imobilidade. Uma alternativa para tornar os protocolos acima mais eficientes, seria combiná-los.

Enquanto os protocolos do tipo *querying* não podem ser ajustados para diferentes características de mobilidade de objetos móveis, o *reporting* não considera a taxa de consulta e a precisão requisitadas pelas aplicações. Com uma combinação de protocolos, no qual se integra o *distance-based reporting* e o *cached querying*, ambas características podem ser alcançadas. Similar ao *distance-based reporting*, o cliente deverá enviar uma mensagem de atualização ao servidor para obter uma dada precisão espacial *d*. Se por ventura a informação armazenada no servidor não satisfizer ao nível de precisão exigido por uma determinada consulta, então o servidor enviará uma mensagem para a fonte a fim de obter sua localização atual como ocorre no *cached querying*.

4.1.3 *QueryService* - Componente de Consultas

O FRAGIL possui a capacidade de responder a uma variedade de questionamentos com relação ao posicionamento dos objetos que estão registrados, assim um serviço de consultas é suportado através de um componente do *framework* denominado *QueryService* e disponibilizado aos usuários através de uma API específica, à qual chamamos *ApplicationAPI*. A seguir descreveremos as consultas que podem ser realizadas através da *ApplicationAPI.*

Uma questão trivial neste tipo de sistemas é determinar a posição atual de um objeto móvel. Aqui, utilizamos uma funcionalidade chamada `positionQuery`, cujo parâmetro é o identificador de um *mo* e a resposta é a posição deste no instante atual. Adotamos que a posição de um *mo* é dada por um sistema de coordenadas tridimensionais, sendo constituído pela tríade x, y e z.

```
positionQuery(moId) → position
```

Em casos em que a comunicação com *mo* estiver indisponível, o usuário poderá indicar se irá receber uma informação nula ou se receberá a última posição atualizada pelo objeto em questão. Uma variação dessa funcionalidade é a possibilidade de se verificar a posição de um objeto em relação a um tempo *t* no passado, tal serviço é disponibilizado através de `temporalPositionQuery`, definido como segue:

```
temporalPositionQuery(moId, Time) → position
```

onde, *moId* refere-se à identificação do objeto móvel, Time é um objeto que representa a informação do momento ao qual se deseja saber a localização do *mo*, representada por *position*. Assim como na funcionalidade anterior, o usuário poderá definir qual resposta ele deverá obter caso não haja registro da posição para o momento requisitado.

Outro tipo de consulta que pode ser oferecida é aquela em que o cliente informa uma área geográfica como parâmetro e obtém como resposta todos os objetos que estão atualmente inseridos dentro da região especificada. Tal consulta está definida como:

```
regionQuery(Area) →{(moId, position)}*
```
[8]

Estendendo a definição acima, podemos querer saber quais os objetos que encontravam-se dentro de uma determinada área em um tempo passado *t*.

```
temporalRegionQuery(Area,Time) →{(moId, position)}*
```

Uma informação bastante valiosa quando estamos monitorando objetos móveis, seria conhecer a trajetória destes, isto é, obter o conjunto de posições ocupadas pelo objeto em relação a um intervalo de tempo determinado. Para tanto, definimos uma consulta denominada `pathQuery`, nela são passados como parâmetros a identificação do objeto

[8] Nas definições das consultas disponibilizadas pelo FRAGIL, estaremos utilizando a seguinte notação: a ocorrência de um elemento zero ou mais vezes está representada pelo símbolo *; a ocorrência de um elemento uma ou mais vezes está representada pelo símbolo +.

rastreado e o intervalo de tempo desejado, representado pelo momento inicial e momento final do rastreamento.

```
pathQuery(moId,Time,Time) →(path)*
```

A consulta acima, deverá retornar uma lista de caminhos que foram percorridos pelo objeto durante o intervalo de tempo solicitado. Cada componente dessa lista, a qual chamamos de `path`, é composto por um ou mais pares representando uma posição no tempo.

```
path→{(Time,position)}+
```

Devido a uma série de fatores o objeto pode não ter sido monitorado durante todo o intervalo de tempo indicado na consulta. Por esse motivo, a união das trajetórias retornadas pela consulta pode não representar um caminho contínuo percorrido pelo objeto.

Por fim, através da funcionalidade `proximityQuery`, os clientes poderão verificar a proximidade de objetos com relação a uma posição específica ou com relação a posição de um outro objeto móvel, podendo ainda definir qual a quantidade de objetos é adequada para os propósitos da consulta.

```
proximityQuery(position,number)→{(moId,dist)}*
```

```
proximityQuery(moId,number)→{(moId,dist)}*
```

Segundo Chon e Agrawal (2002), a estimativa para a determinação de uma localização em tempo futuro leva em consideração o fato de que os objetos móveis não se comportam como partículas, isto é não se movem aleatoriamente no espaço. Assim, tais objetos seguem rotas geralmente influenciadas por limitadores físicos, como por exemplo: uma malha viária onde os limitadores são as estradas pelo qual um carro pode seguir. Segundo esse conceito, este tipo de consulta não poderia ser feita somente com acesso ao banco de dados da localização dos objetos móveis, sendo necessário o conhecimento do banco de dados da aplicação. Tal fato, faz com que o *framework*, pelo menos a princípio, não

suporte consultas para a determinação de uma localização em tempo futuro, uma vez que em sua arquitetura a base de dados da localização encontra-se separada da base de dados da aplicação, não possuindo, ainda, um mecanismo de comunicação do *framework* com essa última base de dados.

4.1.4 *EventService* - Componente de Eventos

Outro componente do *framework* é o registro de eventos baseados na posição dos objetos móveis em relação a outros *mo* ou em relação a uma área geográfica. Inicialmente, o servidor de aplicações (*LBSS*) poderá, através da *ApplicationAPI,* registrar os eventos denominados *RegionEvent* e *DistanceEvent* no *LS*. Ao registrar um *RegionEvent* o usuário será notificado quando o objeto especificado entrar ou sair de uma área pré-determinada. Já no *DistanceEvent* a notificação é enviada quando um objeto especificado estiver a uma certa distância de outro *mo*. Segue a definição dos eventos anteriormente citados:

```
RegionEvent(moId , Area) → notification

DistanceEvent(moId, moId, distance) → notification
```

4.2 Modelagem e Implementação do *Framework*

Nesta seção serão feitos o levantamento e a descrição dos requisitos funcionais, em seguida serão definidas, de maneira clara e concisa, as classes que compõem o *framework*, assim como os relacionamentos estáticos existentes entre as mesmas. Dando continuidade ao processo de modelagem, será definida a colaboração existente entre as classes para a realização das funcionalidades do *framerwork*. Por fim, será mostrada a definição do modelo de dados utilizado. O processo de modelagem do FRAGIL foi regido com base na abordagem

orientada a objetos e descrita pela linguagem UML (*Unified Modeling Language*) (FOWLER, 2000).

4.2.1 Casos de Uso

A modelagem do diagrama de casos de uso é uma técnica usada para descrever e definir os requisitos funcionais de um sistema (MEDEIROS, 2004). O modelo de casos de uso é formado por um diagrama usado para identificar como o sistema se comporta nas várias situações que podem ocorrer durante suas operações. Os componentes deste diagrama são os "atores" (representam qualquer entidade que interage com o sistema) e os "casos de uso" (representam todas as situações possíveis de utilização do sistema).

Na Figura 4.2 apresentamos o diagrama de casos de uso para quando o ator envolvido for o usuário móvel.

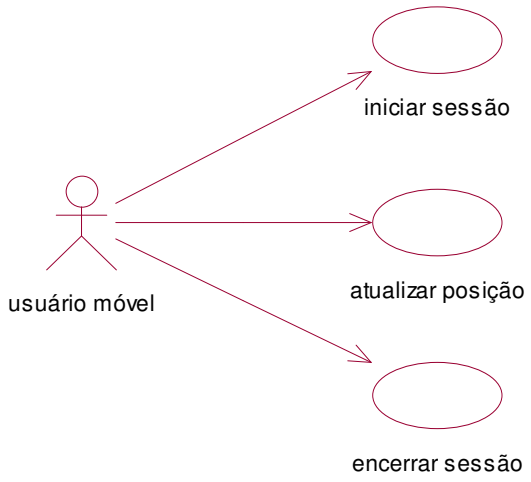

Figura 4.2 Diagrama de Casos de uso para o ator usuário móvel

O caso de uso Iniciar sessão de monitoramento está relacionado ao registro ou cadastro do *mo* no *LS*, o qual manifesta o interesse e conseqüentemente envia a autorização

para que a sua posição seja gerenciada. O mecanismo para a solicitação de registro é fornecido a partir de uma API disponibilizada no próprio *mo*. Efetivado o registro, uma sessão de monitoramento é aberta para o *mo*.

Atualizar posição é o caso de uso no qual o usuário móvel envia sua informação posicional para o servidor de localização levando em consideração um protocolo de atualização. Esta operação só poderá ser realizada se existir uma sessão de monitoramento aberta para este usuário.

Encerrar sessão de monitoramento tem o objetivo de finalizar um ciclo de gerenciamento da informação de localização, isto é, o usuário comunica ao servidor de localização que a partir desse momento não deve mais constar na lista de objetos monitorados e, portanto, não enviará mais informações sobre sua posição geográfica. Se por ventura o usuário desejar voltar a ser rastreado, deverá reiniciar o ciclo registrando-se novamente no *LS*.

Analisando os casos de uso da AplicaçãoLBSS (Figura 4.3), podemos verificar que esse ator, através de uma API, tem a possibilidade de registrar eventos baseados na posição dos *mos* e conseqüentemente receber uma notificação quando as premissas para a ocorrência de tais eventos forem verdadeiras.

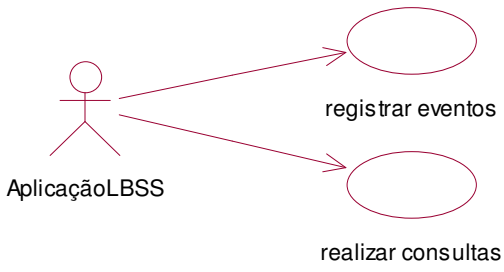

Figura 4.3 Diagrama de Casos de uso para o ator Aplicação LBSS

Um outro caso de uso é o de Realizar consultas, em que a aplicação LBSS pode realizar um conjunto de consultas referentes ao posicionamento dos objetos que estão sendo gerenciados. Essas consultas são disponibilizadas através de uma API específica.

4.2.2 Diagrama de Classes

Feita a definição do diagrama de casos de uso, podemos definir o diagrama de classes do sistema, que demonstra a estrutura estática de suas classes.

Nesta seção, apresentamos o diagrama de classes do FRAGIL. A fim de dar uma visão geral de como as classes estão organizadas e facilitar a visualização de seus relacionamentos, apresentamos na Figura 4.4 um diagrama de pacotes, para, a seguir, descrevermos as classes que constituem cada um desses pacotes.

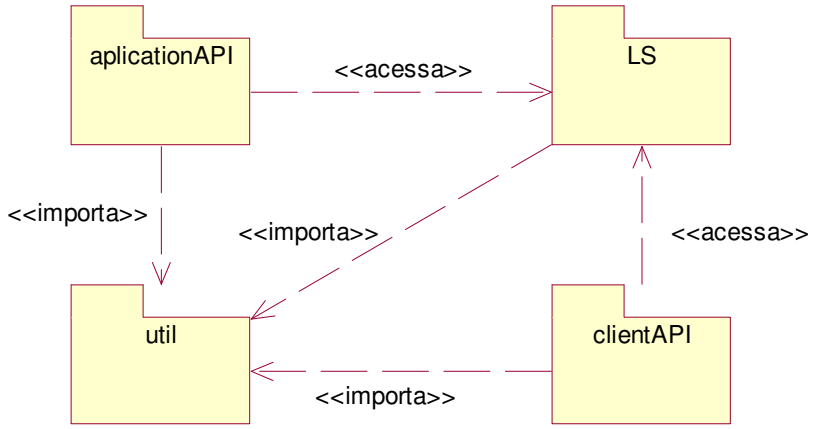

Figura 4.4 Diagrama de Pacotes

4.2.2.1 Pacote util

As classes contidas no pacote útil (Figura 4.5) definem a abstração de elementos fundamentais que serão utilizados em todas as partes do sistema, facilitando o processamento e a troca de informações entre as mesmas. De outra forma, poderíamos dizer que são classes de uso geral.

A classe *Position* representa uma informação posicional e possui como atributos as variáveis reais x, y e z, que guardam a posição em um sistema de coordenadas *ad hoc*. Dois

métodos e um construtor contidos nesta classe merecem destaque, são eles: `toString` e `calcDistance`. O método `toString` gera uma seqüência de caracteres, em um formato específico, que representam uma determinada posição. O formato da seqüência gerada é dado por "x:y:z". Um dos métodos construtores de *Position* possui a capacidade de receber como entrada uma seqüência de caracteres como especificado por `toString` e analisá-la e convertê-la em um objeto. Esse mecanismo facilita muito no tratamento da informação e na transferência de dados.

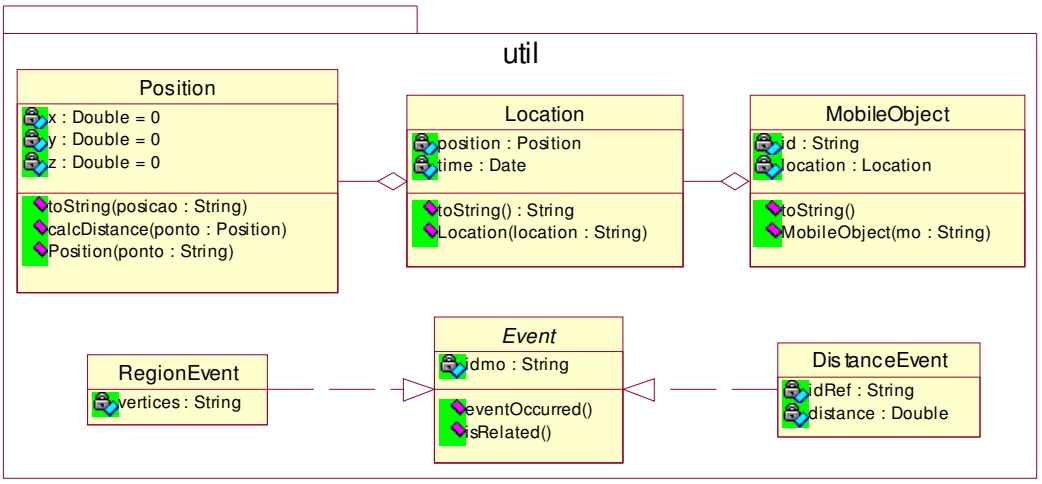

Figura 4.5 Diagrama de Classes do Pacote util

A posição na qual um objeto móvel se encontra em um dado instante é representada pela classe *Location*. Esta possui como atributos a posição e uma data.

Os *mos* são encapsulados por objetos da classe *MobileObject*, cada *mo* é conhecido de forma única dentro do universo da aplicação por um identificador, aqui representado pelo atributo `id`. Todos os *mos* que estão sendo monitorados possuem associados aos mesmos uma localização (*Location*). Assim como na classe *Position*, as classes *Location* e *MobileObject* também possuem a propriedade de criar um objeto a partir de uma *string* no formato gerado por seus respectivos `toString`.

Uma super classe é definida com o intuito de determinar as funcionalidades essenciais para a manipulação e detecção da realização de um evento. A interface *Event* é a base para a criação de eventos dentro do *framework*, ela é constituída pelo atributo `idmo` que identifica a qual objeto móvel o evento está associado e pelos métodos `isRelated` e `eventOccurred`. O primeiro deve determinar se um *mo* que é passado como parâmetro está associado ao evento, já o último encapsula toda a lógica de verificação das premissas para a ocorrência de um evento.

Atualmente, as classes *RegionEvent* e *DistanceEvent* implementam a interface *Event*. A classe *RegionEvent* possui como atributo adicional uma *string* representando os vértices do polígono que representa uma área a ser monitorada. Por sua vez, *DistanceEvent* guarda ainda as informações do *mo* (`idRef`) que se quer monitorar a distância, representada por `distance`.

4.2.2.2 Pacote LS

Pode ser considerado como o núcleo do *framework*, englobando suas principais classes. Esse pacote, mostrado na Figura 4.6, é constituído por um conjunto de classes que possuem, de um modo geral, os métodos responsáveis por receber, tratar e encaminhar as mensagens de solicitação de serviços enviadas pelos diferentes usuários, isto é pelos objetos móveis e aplicações LBS. Possui, ainda, uma classe que realiza a interação com o banco de dados, a *DBLocation*.

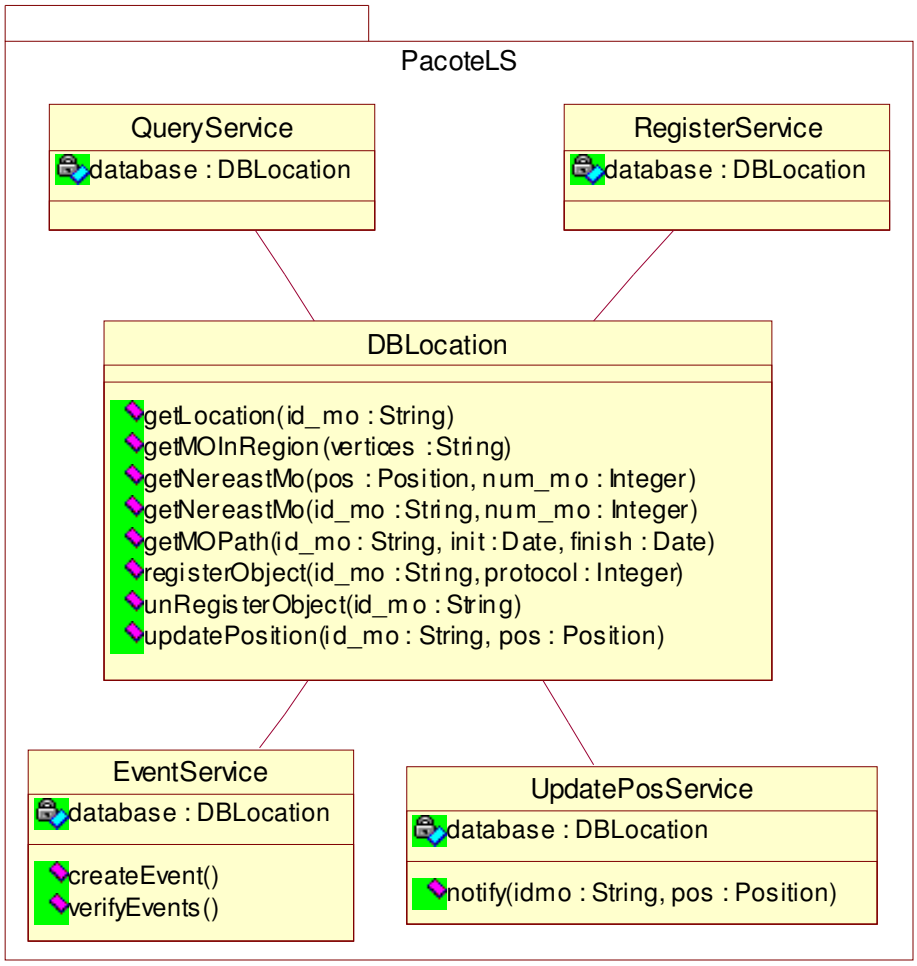

Figura 4.6 Diagrama de Classes do Pacote LS

A classe *QueryService* é a responsável por receber as solicitações de consultas enviadas pela *QueryAPI.* Assim como as demais classes de serviço descritas nesta seção, a *QueryService* estende as funcionalidades de um servidor *Web*, tendo com isso a capacidade de receber e responder a mensagens enviadas pelos usuários através do protocolo Http. Ao receber uma mensagem, ela identifica qual tipo de consulta deve ser executada e, baseado nesse tipo, invoca o método da classe *DBLocation* apropriada. Com um comportamento muito semelhante, temos a classe *RegisterService* que é a responsável por receber as solicitações de registro enviadas pela *AplicationAPI.* Ao receber a mensagem, o *RegisterService* invoca os

métodos da classe *DBLocation* `registerObject e unregisterObject`, dependendo do tipo de mensagem recebida.

A classe *EventService* além de possuir os métodos básicos para o recebimento e envio de mensagens, inclui ainda o método `createEvent` e `verifyEvents`. O primeiro adiciona um novo evento à lista de eventos gerenciáveis, o segundo percorre a lista de eventos verificando se algum deles foi satisfeito. Por outro lado, a classe *UpdatePosService* possui como objetivo receber a informação de atualização dos *mos* e notificar (método `notify`) essa atualização ao *EventService* para a verificação da ocorrência de eventos.

Finalizando as classes do pacote *LS*, temos a classe *DBLocation*, que possui métodos que efetivamente buscam as informações na base de dados necessárias para a realização das principais funcionalidades do *framework*.

4.2.2.3 Pacote ApplicationAPI

Este pacote engloba as classes que possibilitam que as aplicações LBS se comuniquem com o servidor de localização (LS) oferecendo aos seus usuários finais um conjunto de serviços de consultas e eventos sobre a informação de localização dos usuários móveis (Figura 4.7).

A *QueryAPI* disponibiliza a realização de diversas consultas que levam em consideração a localização do usuário, como apresentado na seção 4.1.3.

A *EventAPI* permite a criação de dois tipos de eventos: o *RegionEvent* e o *DistanceEvent*, apresentados na seção 4.1.4. Neste pacote, vale ressaltar a existência da classe *EventListener* que possui a incumbência de verificar se alguma notificação de evento foi enviada pelo servidor de localização. Um objeto dessa classe é criado no momento em que um objeto *EventAPI* for instanciado.

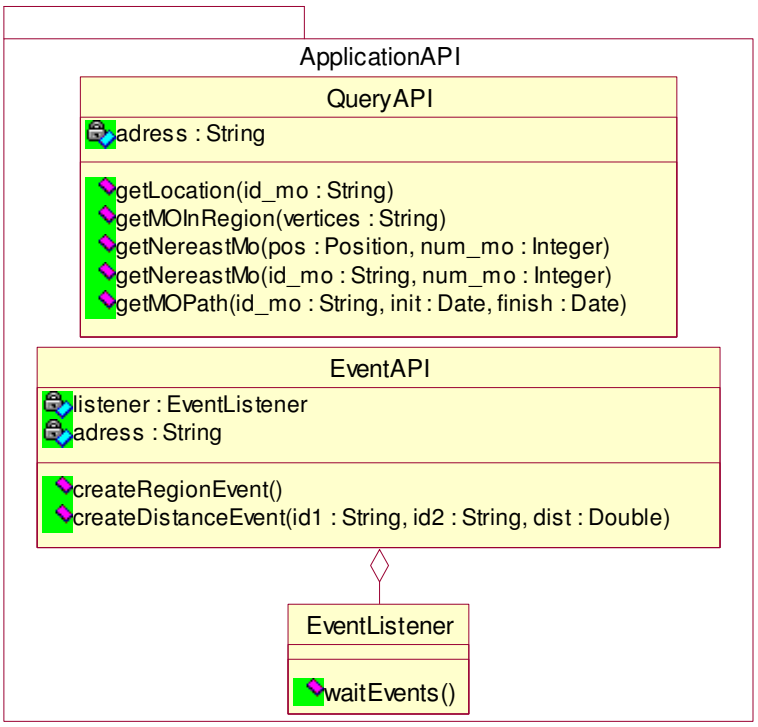

Figura 4.7 Diagrama de Classes do Pacote AplicationAPI

4.2.2.4 Pacote ClientAPI

É composto por apenas uma classe abstrata, a *ClientAPI* (Figura 4.8). Esta classe é a interface de comunicação do usuário móvel (*mo*) com o servidor de localização (*LS*). As operações disponibilizadas nessa classe incluem:

- solicitar o registro e a abertura de uma sessão de monitoramento para o *mo* no *LS*. Para tanto, o usuário deverá utilizar o método `register` passando como parâmetros a identificação do *mo* e o protocolo de atualização a ser utilizado;
- solicitar o fechamento de uma sessão de monitoramento. Nesse caso, basta chamar o método `unregister`, indicando a identificação do *mo*;
- enviar uma mensagem informando ao *LS* a nova posição do objeto móvel. O método `sendPosition` é o responsável por essa operação. Ele, através de

uma conexão Http com o servidor de localização, envia a identificação do *mo* (`id_mo`) e as coordenadas da posição de acordo com o protocolo de atualização que foi definido pelo usuário no momento da criação do objeto do tipo que implementa *ClientAPI*.

A implementação do método `getPosition` fica sob a responsabilidade de quem for implementar a aplicação do cliente móvel. Tal método tem como função obter a posição atual do usuário independentemente da forma que será utilizada para detectar tal posição. O método deverá retornar um objeto do tipo *Position*, que representará as coordenadas de localização do usuário.

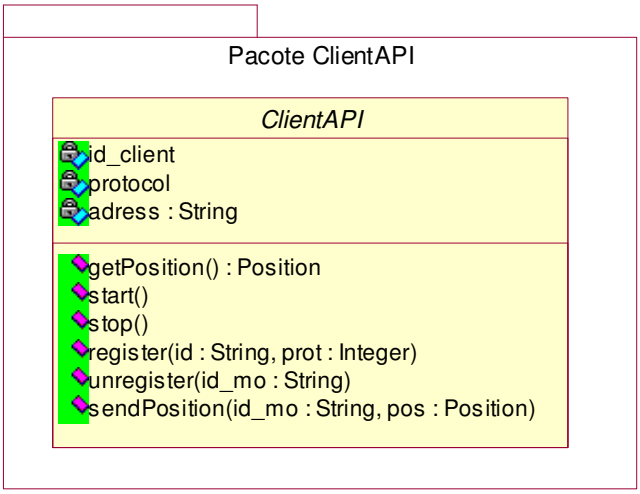

Figura 4.8 Classe ClientAPI

4.2.3 Diagrama de Seqüência

Na subseção 4.2.1 foi mostrada, através do Diagrama de Casos de Uso, a funcionalidade do sistema de uma forma geral. A seguir, será utilizado um diagrama de seqüência para uma análise mais detalhada da colaboração entre os diferentes componentes do *framework*.

Em um primeiro momento será feita uma descrição do diagrama de interação para os casos de uso iniciar/encerrar sessão de monitoramento e atualizar posição (Figura 4.9).

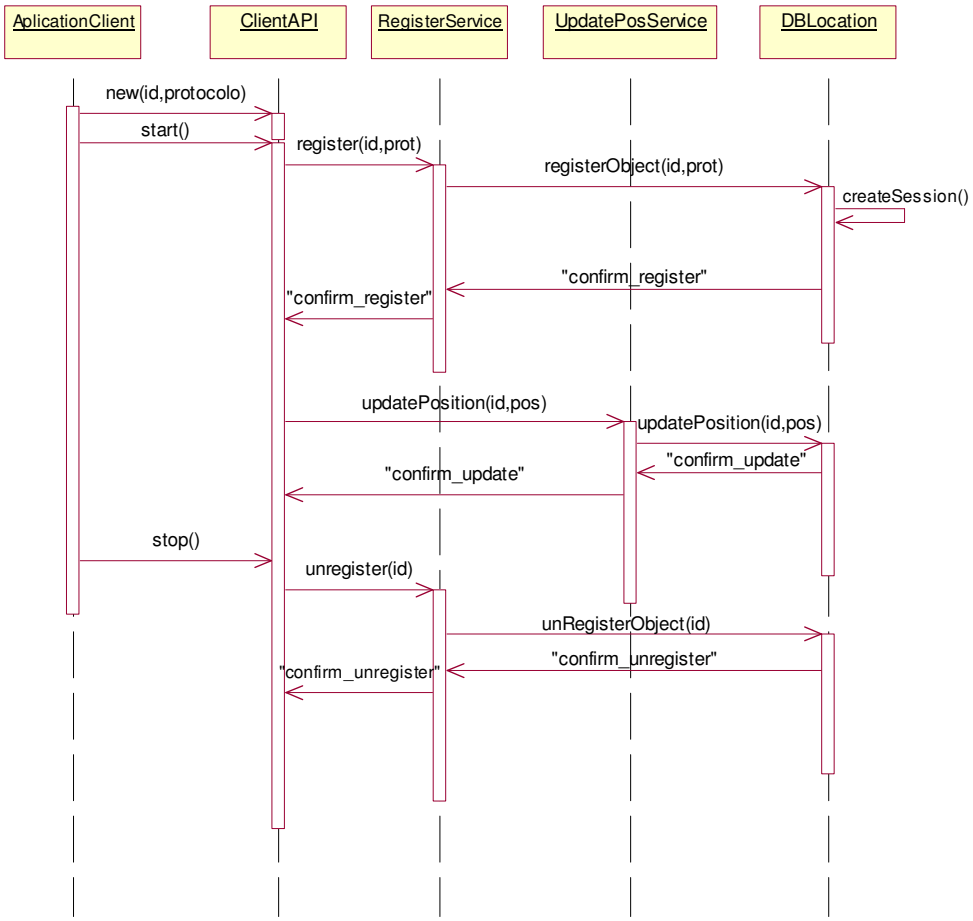

Figura 4.9 Diagrama de Seqüência – Iniciar/Encerrar sessão e atualizar posição

O processo de registro do usuário móvel no servidor de localização ocorre da seguinte forma: uma aplicação qualquer do cliente móvel que queira interagir com o servidor de localização deverá, primeiramente, instanciar um objeto do tipo *ClientAPI*. Este, por sua vez, recebe como parâmetros de criação a identificação do *mo* e um código que representa o protocolo de atualização da informação de localização que será utilizado para a comunicação com o *LS*. Nesse momento, o objeto *ClientAPI* já foi criado e está apto a ser utilizado.

Para dar início ao processo de registro a *AplicationClient* deverá invocar o método `start` da API iniciando o ciclo de vida da *thread*. Ao ser iniciada, a *ClienteAPI* envia, através do protocolo Http, uma mensagem do tipo GET contendo como parâmetros a identificação do *mo* e o protocolo que o mesmo irá utilizar.

Ao receber uma solicitação de registro, o *RegisterService* faz o tratamento dos parâmetros contidos na mensagem e repassa essa informação para o *DBLocation* através do método `registerObject`. Nesse instante, é feita uma verificação da existência de uma sessão para o usuário, a mesma se existir é encerrada, já que seria fruto de um erro no fechamento da sessão aberta anteriormente, causado pela desconexão temporária, característica ainda não tratada pelo *framework*. Feitas as verificações necessárias, inicia-se o processo de criação da sessão de gerenciamento. Criar uma sessão significa inserir uma nova tupla na tabela de sessões do banco de dados de localização com um *status* indicando que a mesma está em aberto.

Por fim, uma mensagem informando o resultado da operação é retornado ao *RegisterService*, que transforma a informação recebida em uma mensagem que possa ser interpretada pela *ClientAPI* e a envia para a mesma. A *ClientAPI* analisa a resposta enviada, se receber um “register_ok” o processo de atualização de posição é invocado, caso contrário uma mensagem de erro é enviada para a AplicationClient e o ciclo de vida da ClientAPI é encerrado.

Depois que um *mo* for devidamente registrado, inicia-se o processo de atualização da posição de localização baseado no protocolo escolhido. Constantemente, a *ClientAPI* executa um método denominado `getPosition`, que obtém a posição do *mo* a partir de um PDE. Caso essa localização satisfaça às condições determinadas por cada protocolo, então uma mensagem de atualização da posição é enviada para o *LS* através do método

`sendPosition`. UpdatePosService faz o tratamento da informação dos parâmetros contida na mensagem e repassa essa informação para o *DBLocation* através do método `updatePosition`. Nesse ponto, é feita a inclusão de um registro na tabela que armazena as informações da localização do *mo*, denominada obj_moveis. Para cada nova atualização é criada uma linha contendo a identificação do objeto, as coordenadas de localização e o tempo em que as mesmas foram obtidas. Uma mensagem informando o resultado da operação é retornada ao *UpdatePosService*, este transforma a informação recebida em uma mensagem que possa ser interpretada pela *ClientAPI* e a envia para a mesma. Este processo de atualização se repete enquanto a sessão de gerenciamento da informação de localização estiver aberta para o *mo*.

O processo de encerrar uma sessão é iniciado através da chamada do método `stop` pertencente à *ClientAPI* pela *AplicationClient*. O processo de atualização de posição é encerrado e uma mensagem contendo como parâmetro a identificação do *mo* é enviada ao *RegisterService*. Este verifica se a mensagem é para a criação ou finalização de uma sessão, faz o tratamento da informação dos parâmetros contida na mensagem e repassa essa informação para o *DBLocation* através do método `unregisterObject`.

Nesse momento, é chamado o método `closeSession`. Fechar uma sessão significa: atualizar o campo *status*, indicando que a mesma está encerrada, em um registro de sessão na tabela de sessões do banco de dados de localização; mover todas as informações de posicionamento, referentes ao *mo* em questão, contidas na tabela obj_moveis para uma tabela que armazene as informações históricas das posições dos *mos* no decorrer do tempo (obj_moveis_hst). Por fim, uma mensagem informando o resultado da operação é retornado ao *RegisterService*, que transforma a informação recebida em uma mensagem que possa ser interpretada pela *ClientAPI* enviando-a para a mesma e encerrando o ciclo de vida da *ClientAPI*.

Na Figura 4.10, é mostrado o diagrama de interação para o caso de uso realizar consultas. A *QueryAPI* disponibiliza um conjunto de consultas referentes ao posicionamento dos objetos que estão sendo monitorados pelo *LS*. Para que uma aplicação LBS possa coletar informações a respeito da localização dos usuários móveis que fazem parte do sistema, deverá instanciar um objeto do tipo *QueryAPI* passando o endereço do *LS*.

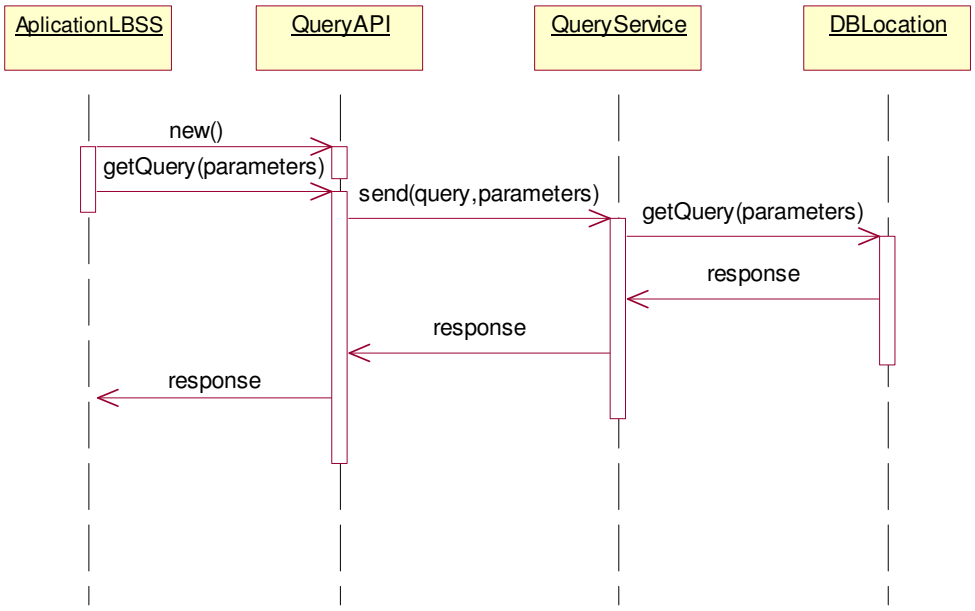

Figura 4.10 Diagrama de Seqüência – Realizar Consulta

Atualmente, como descrito na seção 4.1.3, são disponibilizadas pelo FRAGIL as seguintes consultas: *positionQuery*, *regionQuery*, *pathQuery*, *proximityQuery*, *temporalRegionQuery* e *temporalPositionQuery*. Cada uma delas recebe um conjunto de parâmetros distintos, bem como retornam como resposta objetos dos mais variados tipos. Para de fato realizar uma consulta, a *ClientLBS* deverá executar o método `getQuery`, pertencente à classe *QueryAPI*, passando o tipo de consulta a ser realizada e os parâmetros típicos de cada uma. A *QueryAPI* envia, através do protocolo Http, uma mensagem do tipo GET contendo como parâmetros as informações anteriormente citadas.

Ao receber uma solicitação de consulta, o *QueryService* identifica o tipo de consulta, faz o tratamento da informação dos parâmetros contida na mensagem e repassa essa informação para o *DBLocation* através do método de consulta apropriado. O *DBLocation*, por sua vez, coleta as informações nas tabelas de localização e uma mensagem informando o resultado da operação é retornado ao *QueryService*, que encapsula a resposta recebida em uma mensagem que possa ser manipulada pela QueryAPI e a remete para a mesma. Essa mensagem é recebida pelo método `getQuery`, nele as informações são encapsuladas em objetos de acordo com o tipo de retorno e devolvidos para a *ClientLBS*.

A aplicação LBS também possui a capacidade de registrar eventos baseados na posição dos objetos móveis para que o *LS* monitore e envie uma notificação quando as premissas para a ocorrência de tais eventos forem verdadeiras. A atual versão do *framework* permite a verificação da ocorrência dos seguintes eventos: *regionEvent* e *distanceEvent*, como mostrado na seção 4.1.4.

O processo de cadastro de um evento (Figura 4.11) inicia-se quando a aplicação LBS (*AplicationLBSS*) instancia um objeto do tipo *EventAPI*, passando como parâmetro de inicialização o endereço do servidor onde está localizado o *LS*. Ao ser criada, a *EventAPI* instancia um objeto do tipo *EventListener*. Como o próprio nome sugere, este objeto, através do método `waitEvents`, caracteriza-se por ficar permanentemente verificando a chegada de alguma notificação de eventos advinda do servidor de localização.

O próximo passo é registrar, de fato, um determinado evento. Isso ocorre pela execução do método *registerEvent* definido pelo objeto *EventAPI*. Como no caso das consultas, cada evento recebe um conjunto de parâmetros distinto, por essa razão é passado para o método `registerEvent` parâmetros com as seguintes informações: o tipo do evento a ser registrado, identificação do objeto móvel associado ao evento e os parâmetros próprios de cada um.

A *EventAPI* envia, através do protocolo Http, uma mensagem contendo como parâmetros as informações anteriormente citadas. Ao receber uma solicitação de cadastro de evento, o *EventService* identifica o tipo de evento, faz o tratamento da informação dos parâmetros contida na mensagem e cria um novo objeto do tipo *Event* de acordo com o tipo especificado pelo usuário. Em seguida, uma mensagem indicando o sucesso ou o fracasso da operação de registro é enviada para a *EventAPI*, que a remete para a *AplicationLBS*.

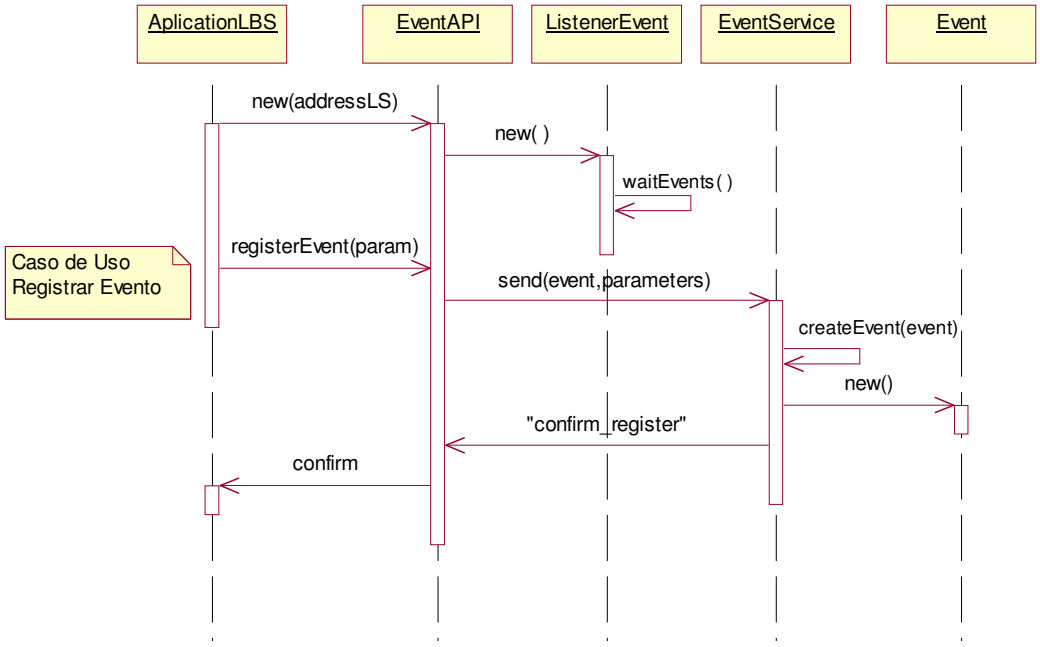

Figura 4.11 Diagrama de Seqüência – Registrar Evento

Feito o registro no *LS*, resta-nos saber como é detectada a ocorrência de um evento e gerada a notificação para o *EventListener*, situação ilustrada na Figura 4.12. Para os nossos propósitos, a fonte geradora do evento está na mudança da localização de um objeto móvel. Por conta disso, sempre que uma mensagem de atualização de posição for recebida pelo *UpdatePosService*, este deverá enviar a identificação do objeto e sua nova posição para o *EventService*, que, além de registrar os eventos, é responsável por gerenciar a ocorrência deles.

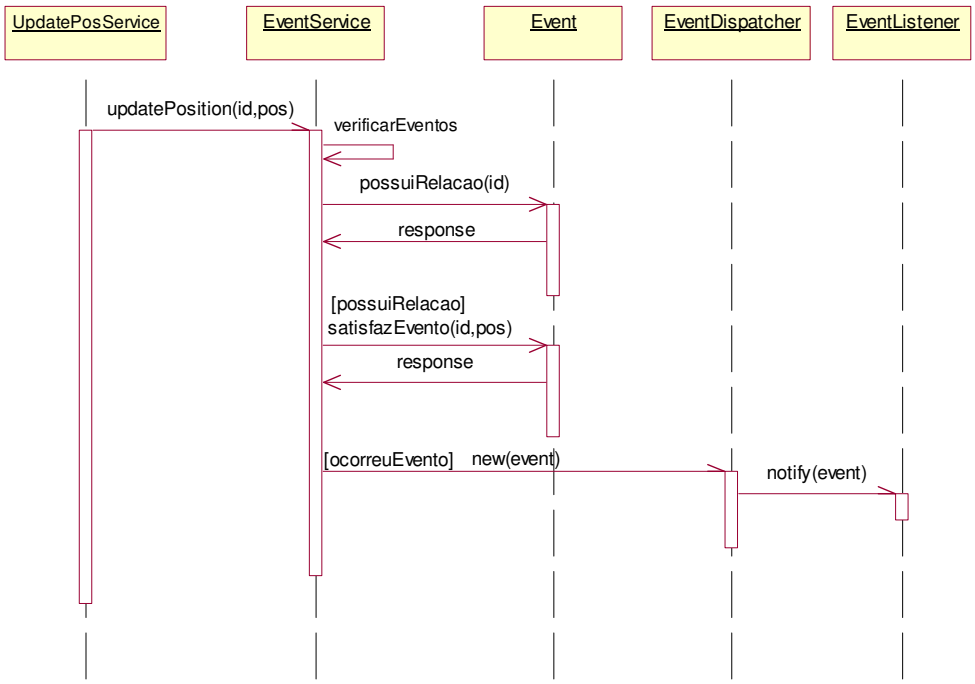

Figura 4.12 Diagrama de Seqüência – Notificar Evento

Ao receber uma informação de atualização da posição, para cada evento que esteja cadastrado é feita uma verificação se o *mo* está relacionado ao evento. Essa verificação é obtida através do método `possuiRelacao` do *Event*. Caso o *mo* esteja relacionado ao evento cadastrado, então o método `satisfazEvento` é executado de forma que verifique se as condições determinadas para o evento se tornaram verdadeiras, acarretando na ocorrência do mesmo. Se as premissas tiverem sido satisfeitas, então um objeto do tipo *EventDispatcher* é criado tendo como informação inicial uma instância do evento ocorrido. Para finalizar, o *EventDispatcher* envia uma mensagem de notificação de ocorrência do evento para o *EventListener*.

4.2.4 Modelagem de Dados

Para possibilitar o gerenciamento da informação de localização, o *framework* utiliza um esquema de dados independente da aplicação LBS. Nesta seção estaremos apresentando esse esquema de dados (Figura 4.13).

O esquema de dados do FRAGIL envolve um conjunto de tabelas que tem por finalidade armazenar as informações de localização dos usuários ao longo do tempo, permitindo a administração dessas informações através da manipulação de sessões.

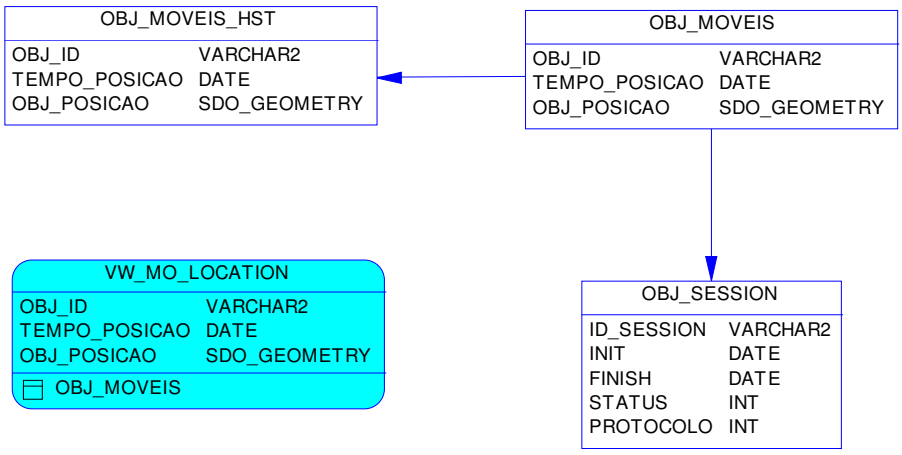

Figura 4.13 Diagrama de Dados

A tabela OBJ_SESSION mantém as informações de todas as sessões de monitoramento. No FRAGIL uma sessão representa um espaço de tempo que vai desde o momento em que o objeto passou a ser rastreado até o momento em que a posição do objeto móvel deixa de ser monitorada, seja por vontade do próprio usuário ou por motivos de perda de conexão. Esta tabela é composta por um identificador do objeto móvel (*obj_id*), um identificador da sessão de monitoramento (*id_session*), um campo que contém o momento em que a sessão foi iniciada (*init*), outro que representa o instante em que a sessão foi encerrada

(*finish*), um indicando a situação atual da sessão (*status)* e, ainda, um que identifica o protocolo de atualização a ser utilizado (*protocolo).*

Os dados de posicionamento dos objetos que estão sendo atualmente gerenciados são armazenados na tabela OBJ_MOVEIS. Isto é, enquanto uma sessão de monitoramento para um determinado *mo* estiver em aberto (*status ativo*), suas posições estarão disponíveis nessa tabela. A partir do momento em que um objeto tiver sua sessão de monitoramento encerrada (*status inativo*), as tuplas com as informações de localização desse objeto serão transferidas para uma tabela que contém o histórico da localização, denominada OBJ_MOVEIS_HST. Esta, portanto, contém todas as informações de localização dos objetos que não estão sendo mais gerenciados ou de sessões já encerradas. Tanto a tabela OBJ_MOVEIS, quanto a OBJ_MOVEIS_HST possuem a mesma estrutura formada pelos seguintes campos: identificador do objeto móvel (*obj_id*), data/hora em que a posição foi capturada (*tempo_posicao*) e posição do objeto em um dado momento (*obj_posicao*).

Por fim, com o objetivo de facilitar as consultas a serem realizadas, foi criada uma visão com base na tabela OBJ_MOVEIS, denominada VW_MO_LOCATION. Possuindo a mesma estrutura da OBJ_MOVEIS, representa a localização atual de todos os objetos que estão sendo gerenciados.

5 ESTUDO DE CASO: SISTEMA DE RASTREAMENTO DE USUÁRIOS MÓVEIS (SRUM)

Com o intuito de validar as funcionalidades do *framework* proposto, foi desenvolvida uma aplicação, denominada SRUM, baseada na localização de usuários móveis. A mesma foi concebida para funcionar como uma central de rastreamento dos usuários que estiverem circulando na região do centro histórico da cidade de São Luís. Esta ferramenta disponibiliza, através de uma interface gráfica, recursos de rastreamento, determinação de proximidade e consultas de rotas percorridas pelo usuário, tendo as respostas visualizadas através de um mapa da região.

Em razão da indisponibilidade da infra-estrutura necessária para o desenvolvimento de uma aplicação em um ambiente real, foi adotado o princípio de simulação para o caminho que está sendo percorrido pelos usuários móveis no decorrer do tempo. Contudo, tal medida não afeta de maneira significativa os propósitos da criação desta aplicação, que visa tão somente avaliar o potencial de utilização do FRAGIL.

No decorrer deste capítulo estaremos apresentando todo o processo de desenvolvimento da aplicação, ressaltando suas funcionalidades, limitações e a interação com as API's fornecidas pelo FRAGIL.

5.1 Descrição das Funcionalidades

Nesta seção serão apresentados os aspectos de funcionalidade do sistema, determinando "o que" o sistema faz. Para descrever e definir esses requisitos funcionais iremos utilizar um diagrama de casos de uso complementado por informações textuais.

O SRUM divide-se em dois módulos funcionais: o cliente e a ferramenta de monitoramento. O primeiro está localizado no dispositivo móvel do usuário, é representado

por um programa que, através da API do *framework*, fornece a informação de localização do cliente ao servidor, dando ao usuário a possibilidade de decidir, no momento do registro, se quer ou não ser rastreado. Já a ferramenta de monitoramento é uma aplicação *desktop* que se utiliza das API's de consulta e eventos para dar aos seus usuários uma visão ampla da distribuição geográfica dos clientes, cuja posição esteja sendo acompanhada.

A divisão em módulos funcionais cria naturalmente a presença de dois atores para o sistema, cada qual interagindo com um módulo distinto. A Figura 5.1 mostra os diagramas de casos de uso para o SRUM, onde o ator usuário móvel refere-se ao módulo cliente e o ator *admin* representa o usuário da ferramenta de monitoramento.

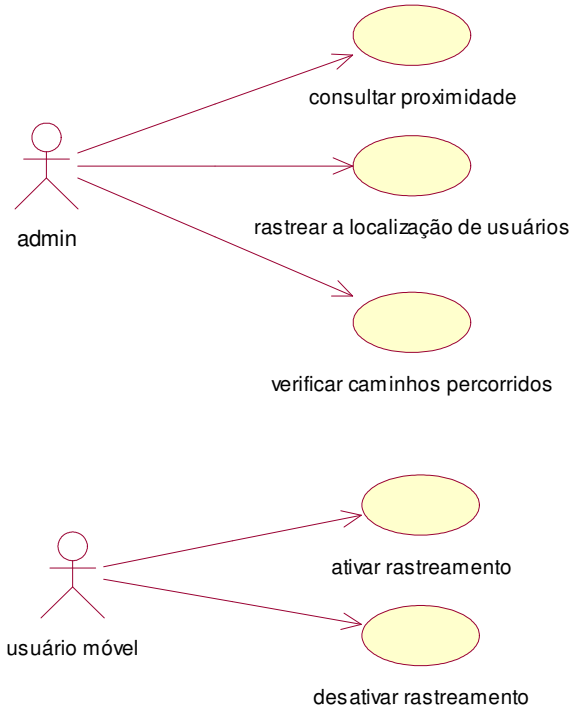

Figura 5.1 Diagrama de Casos de Uso do SRUM

No caso de uso Consultar Proximidade, o usuário da ferramenta de monitoramento tem a possibilidade de obter e visualizar no mapa quais os *N mos* que estão

mais próximos de um determinado ponto de interesse (POI) ou de um outro objeto móvel. Em Rastrear a localização de usuários, o *admin* pode acompanhar a atual posição dos clientes situados na região do centro histórico, visualizando essas informações por meio de um mapa da área. Outro caso de uso do ator *admin* é Verificar caminhos percorridos, neste uma consulta temporal poderá ser feita para obter a rota percorrida por um cliente durante um intervalo de tempo pré-determinado. A rota é visualizada de forma gráfica no mapa.

Para o ator usuário móvel temos apenas dois casos de uso. No Ativar rastreamento, levando em consideração a questão da privacidade, o usuário móvel pode determinar se quer ou não que sua localização seja conhecida por terceiros. Ao ativar rastreamento, o programa passa a enviar temporariamente informações sobre a posição do usuário ao servidor de localização. Por outro lado, no Desativar rastreamento, o usuário móvel comunica que a partir deste momento não deve mais constar na lista de objetos monitorados e, portanto, não enviará mais informações sobre sua posição geográfica ao servidor. Com isso o cliente deixa de ser rastreado pela ferramenta de monitoramento.

5.2 Implementação

De uma forma geral, podemos dizer que a arquitetura do SRUM é composta por três camadas básicas: camada de serviços, camada de apresentação e camada de dados. A primeira é encapsulada pelo *LS* e pelas APIs do FRAGIL, fazendo o gerenciamento da informação de localização dos usuários móveis. A camada de apresentação é formada pelo módulo cliente e pela ferramenta de monitoramento. Por fim, a camada de dados guarda os pontos de interesse referentes ao centro histórico de São Luís.

A seguir, serão descritos os aspectos de implementação do módulo cliente e da ferramenta de monitoramento, bem como suas interações com as APIs fornecidas pelo *framework*.

5.2.1 Módulo Cliente

O *software* cliente foi desenvolvido para ser utilizado em dispositivos móveis que suportem a tecnologia Java, para tanto foi implementado utilizando a linguagem J2ME[9], com a configuração CLDC 1.1(CLDC, 2005) e o perfil MIDP 2.0 (MIDP, 2005).

O *software* cliente é representado por um MIDlet[10] presente no dispositivo móvel chamado *ClientMIDlet* e por uma classe denominada *ClientLocation* que é uma implementação da classe abstrata *ClientAPI* fornecida pelo *framework* FRAGIL. O *ClientMIDlet* funciona apenas como uma interface gráfica com o usuário, ficando sob a responsabilidade da *ClientLocation* todo o processo de captura, tratamento e envio da informação de localização do dispositivo móvel para o servidor (*LS*).

Na Figura 5.2 são mostradas as telas da aplicação disponibilizada para o cliente. A primeira delas é uma tela de abertura informando o nome e uma breve descrição da utilidade do *software*. Ao selecionar a opção iniciar o usuário é levado para uma segunda tela na qual terá a possibilidade de executar as duas funcionalidades disponíveis, que são: ativar e desativar rastreamento.

Ao selecionar o item *Iniciar Serviço*, uma instância da classe *ClientLocation* é criada iniciando os processos de registro de sessão e envio da posição atual do usuário para o

---

[9] Java 2 *Micro Edition* (J2ME) é a edição da linguagem Java usada no desenvolvimento de aplicações para dispositivos computação portáteis e móveis. J2ME é considerada um subconjunto da edição padrão de Java (J2SE).

[10] São aplicações que executam em dispositivos sobre o perfil MIDP.

servidor de localização. Ao ser iniciada, a classe *ClientLocation* recebe como parâmetros o endereço de rede do *LS* e uma constante indicando o tipo de protocolo de atualização que será utilizado. Em nossa aplicação estaremos utilizando os protocolos *time-based reporting* e o *distance-based reporting*.

Figura 5.2 Telas de interface da aplicação cliente

De outra forma, ao escolher *Encerrar Serviço* é feita uma verificação se o serviço havia sido iniciado. Em caso afirmativo, uma mensagem é enviada ao servidor informando para fechar a sessão de monitoramento desse cliente.

Como dito anteriormente, para se comunicar com o servidor de localização é necessário que se implemente a classe *ClientAPI* do FRAGIL. Implementar essa classe significa escrever o código para o método *getPosition,* que deve conter o mecanismo para capturar a posição do usuário móvel de acordo com as tecnologias de localização escolhidas

(Figura 5.3). Tal método não possui valores de entrada e deve retornar um objeto do tipo *Position* (pacote util).

```
public  class ClientLocation implements ClientAPI {
   ...
  public Position getPosition(){
  //mecanismo de captura da posição do usuário móvel deve ser escrita aqui
  }
   ...
}
```

Figura 5.3 Trecho de código exemplificando a realização da classe ClientAPI.

Em nossa aplicação não foi possível utilizar um equipamento que pudesse fornecer a localização do usuário, tal como um receptor GPS. Portanto, o código do método *getPosition* foi implementado de forma a simular a posição do usuário em determinado instante, para isso, um arquivo foi preenchido com várias posições que equivalem a um caminho que o usuário estaria percorrendo. Esse conjunto de pontos que constituem um caminho percorrido não é georeferenciado, isto é não representa a localização geográfica real do usuário, sendo formado apenas por coordenadas gráficas no contexto da imagem que representa o mapa da região monitorada. A cada chamada de *getPosition* um valor do arquivo é retornado.

5.2.2 Ferramenta de Monitoramento

A Ferramenta de monitoramento tem como objetivo oferecer ao *admin* uma visão completa da posição dos usuários móveis que transitam no centro histórico de São Luís no decorrer do tempo. Para tanto, utiliza-se de uma interface gráfica bastante intuitiva e fácil de ser manuseada. A utilização de um mapa representando a região a ser monitorada possibilita uma representação mais efetiva do ambiente, permitindo que os resultados das consultas se

tornem de fácil absorção pelo consumidor dessa informação. Esse sistema deve funcionar como suporte à tomada de decisões dos usuários, facilitando seu trabalho e tornando mais efetiva sua interação com os dados.

A ferramenta de monitoramento foi desenvolvida na linguagem Java, sendo composta basicamente por uma classe de interface gráfica com o usuário, denominada *PrincipalGUI*, e por uma classe *thread* chamada *Monitor*, que verifica temporariamente quais as opções de gerenciamento escolhidas pelo *admin* e baseadas nelas realiza as consultas no servidor de localização. Para que possa ser efetuada a realização de consultas no servidor de localização é necessária a utilização da classe *QueryAPI* do pacote *AplicationAPI* fornecido pelo FRAGIL. Além disso, a *PrincipalGUI e Monitor* também utilizam as classes do pacote *util*, para encapsularem e manipularem os dados enviados pelo *LS* em decorrência da realização de consultas.

A interface gráfica da ferramenta de monitoramento, mostrada na Figura 5.4, é composta por um *menu* de opções, por uma área para a escolha do gerenciamento a ser realizado e por um mapa representando as ruas e os pontos de interesse do centro de São Luís, bem como os usuários móveis que estão na localizados região. Esse mapa, não está georeferenciado, constituindo-se apenas de uma imagem no formato JPG. Sendo assim, o mapa apenas ilustra a área de interesse a ser monitorada, não possuindo qualquer associação direta com a representação da localização física da área representada.

Na parte superior da interface pode-se observar os menus configuração, sobre e sair. O primeiro permite que o usuário determine o endereço de rede do servidor de localização (*LS*) que será utilizado pela classe *QueryAPI* para a realização das consultas. Outra característica que pode ser configurada através desse *menu* é o intervalo de tempo em que devem acontecer a atualização da informação de localização dos usuários móveis no mapa (*refresh*). Esse valor é medido em segundos, tendo como valor padrão o tempo de 30

segundos. O *menu sobre* é meramente informativo, mostrando a descrição da aplicação, quem desenvolveu e a versão do *software*. Já o *menu sair*, encerra todas as conexões com o LS, se existirem, e finaliza a execução do programa.

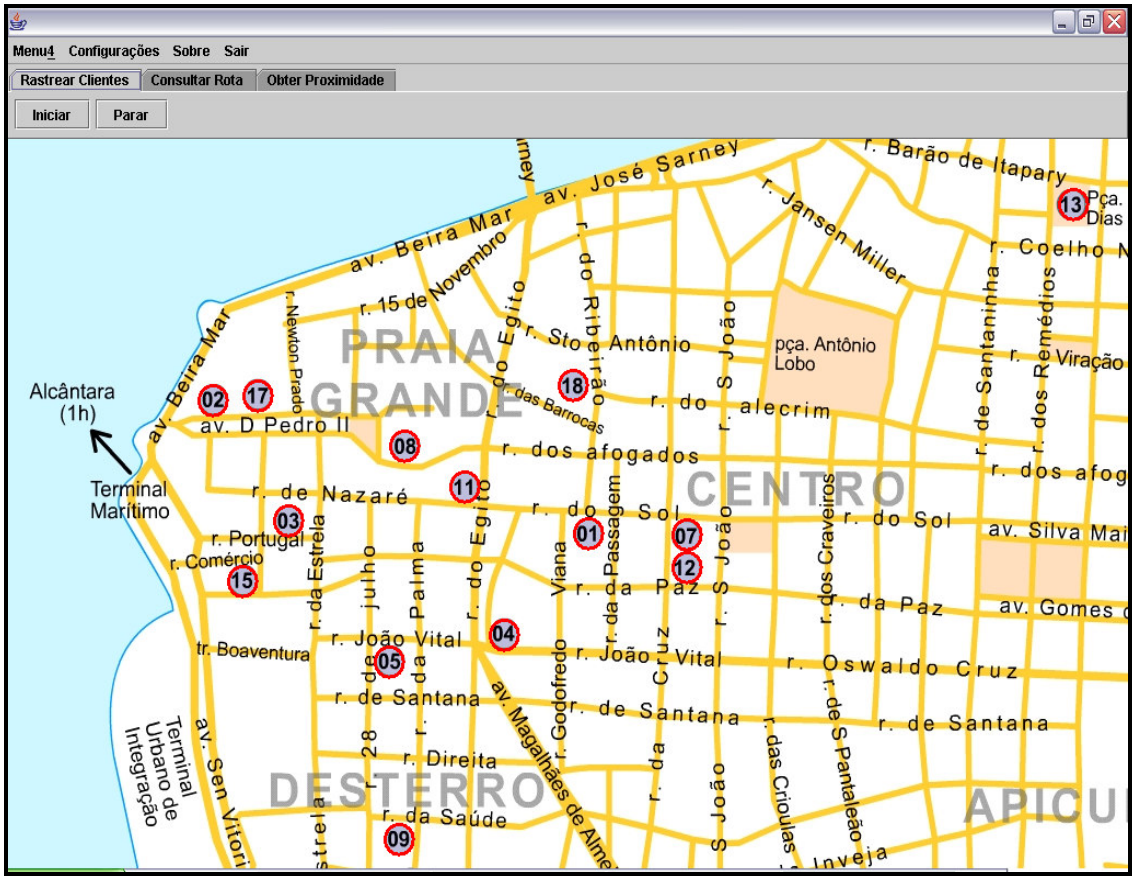

Figura 5.4 Interface Gráfica da Ferramenta de Monitoramento.

Logo abaixo dos menus há uma área para a escolha do tipo de gerenciamento a ser realizado. Existem três tipos de gerenciamento que podem ser escolhidos: *Rastrear Usuários*, *Consultar Percurso* e *Consultar Proximidade,* representando cada um dos requisitos funcionais descritos na seção 5.1. Esses serviços estão dispostos graficamente em forma de fichas, contendo informações que deverão ser preenchidas pelo usuário de acordo com o tipo de gerenciamento escolhido.

Ao selecionar a ficha *Rastrear Usuários* (Figura 5.5), deverá ser informado se o monitoramento se aplica a todos os objetos móveis ou se apenas a um objeto específico. Outros dois botões, um que inicia e outro para o monitoramento, também podem ser visualizados nesta ficha. No momento em que o usuário pressiona o botão *Iniciar*, uma mensagem é mandada ao *Monitor* sinalizando o tipo de consulta que deverá ser realizada. A partir daí, o *Monitor* passa a utilizar, em intervalos de tempo definidos pelo usuário (*menu Configurações*), os métodos da classe *QueryAPI* para obter o atual posicionamento dos usuários móveis.

Figura 5.5 Ficha Rastrear Usuários

Para essa funcionalidade são utilizados dois métodos da *QueryAPI*: *getLocalizacao* e *getMOinRegion*. O primeiro é utilizado para obter a localização de um usuário específico, já o *getMOinRegion* é utilizado para obter a localização de todos os objetos que se encontram em uma região específica, no caso o centro histórico.

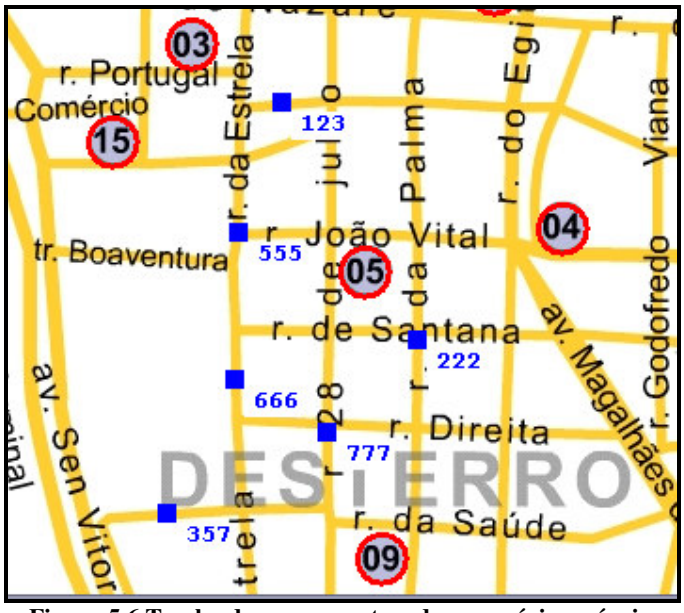

Figura 5.6 Trecho do mapa mostrando os usuários móveis.

A informação de localização obtida pelo *Monitor* é repassada para a *PrincipalGUI* que se encarrega de atualizar a informação disponibilizada no mapa (Figura 5.6). O processo de captura, tratamento e visualização da localização é feito de forma dinâmica, ou seja, enquanto a tecla *Parar* não for pressionada, o usuário sempre terá no mapa a posição mais atual do usuário móvel.

Através da ficha *Consultar Percursos* (Figura 5.7) é possível visualizar qual o caminho percorrido por um usuário móvel em um determinado intervalo de tempo. Para isso, deve-se selecionar a identificação do *mo* e indicar o momento inicial e final. Ao pressionar o botão *Consultar*, o tipo de consulta juntamente com os atributos contidos na ficha são enviados ao *Monitor*, que utiliza o método *getMOPath*, da classe *QueryAPI*, para solicitar ao *LS* a informação do caminho percorrido pelo *mo*. Ao obter a resposta, o *Monitor*, a repassada para a *PrincipalGUI* que desenha o caminho no mapa (Figura 5.8).

Figura 5.7 Ficha Consultar Percursos.

Figura 5.8 Representação de uma rota percorrida por um *mo*.

Os usuários da ferramenta poderão verificar a proximidade de *mo* em relação a um ponto de interesse específico ou com relação à posição de um outro objeto móvel, podendo ainda definir qual a quantidade de objetos é adequada para os propósitos da consulta. Assim, na ficha *Consultar Proximidade* (Figura 5.9), primeiramente deverá ser selecionado qual o tipo de referência será utilizado (POI ou *mo*). Feita a escolha, automaticamente é carregada uma lista dos objetos de referência contendo, respectivamente, a relação dos pontos de interesse do centro histórico ou o conjunto de *mos* que atualmente estão localizados na região. Por fim, o número máximo de objetos a serem visualizados é preenchido.

Figura 5.9 Ficha Obter Proximidade.

O processo de captura, tratamento e visualização dinâmica da informação de localização é realizado de forma similar ao *Rastrear Usuários*, a única diferença é que o método da *QueryAPI* utilizado para obter os *mos* próximos é o *getNereastMo*. Isso quer dizer que enquanto a tecla *Parar* não for pressionada, o usuário sempre visualizará no mapa a posição atual dos *N mos* que se encontram mais próximos do objeto de referência escolhido.

5.2.3 Considerações Finais

O FRAGIL, de uma maneira geral, atendeu às expectativas com relação à sua funcionalidade, portabilidade e flexibilidade. A utilização de suas APIs propiciou aos desenvolvedores das aplicações mostradas neste capítulo uma total abstração do processo de gerenciamento da informação de localização feita pelo *LS*. Tal fato fez com que os esforços no processo de construção do sistema estivessem concentrados apenas nas questões

relacionadas ao domínio da aplicação e na interação dos usuários com a interface gráfica, permitindo um ganho significativo no tempo utilizado para a construção do sistema.

6 CONCLUSÃO

Cada vez mais os usuários sentem a necessidade de ter acesso à informação em qualquer momento e em qualquer lugar. Essa característica revela uma demanda natural de serviços que forneçam informações ao usuário levando-se em conta a sua localização. Neste contexto surge uma categoria de novas aplicações conhecidas como LBS, que permitem aos usuários, através de uma rede sem fio, utilizarem serviços baseados em sua posição geográfica.

No presente trabalho foi apresentado primeiramente o estado da arte dos Sistemas Baseados na Localização, tendo sido definidos a conceituação e a taxonomia utilizadas em tais sistemas, bem como mostrado suas principais aplicações, destacando sua importância diante das atuais necessidades e exigências dos usuários de dispositivos móveis. Foi realizado, ainda, um estudo apresentando a descrição dos diversos tipos de tecnologias, que podem ser utilizadas por aplicações LBS, para a detecção da posição de um determinado usuário, cada uma delas apresentando suas qualidades e suas deficiências em relação a fatores como o custo de implantação, a área de cobertura (*indoor* ou *outdoor*) e, principalmente, a precisão da informação. Em especial, o levantamento dos *frameworks,* das arquiteturas utilizadas e dos requisitos funcionais necessários para a construção dessas aplicações deram sustentação teórica de fundamental importância para a definição dos serviços a serem reutilizados, para a definição das funcionalidades oferecidas, para a definição e construção de uma arquitetura base e para a concepção e implementação do *framework*, objetivos primordiais deste trabalho. Por fim, foi apresentada uma aplicação para a validação do *framework* proposto.

Além das contribuições inerentes de trabalhos de dissertação, como a análise do estado da arte das diversas áreas que envolvem os objetos da pesquisa, destacam-se como as principais contribuições deste trabalho a concepção de uma arquitetura reutilizável para as

funcionalidades de gerenciamento da informação de localização que propicie a utilização de diversos protocolos de atualização dessa informação, a concepção de um *framework* baseado nessa arquitetura e sua implementação.

Os *frameworks* se apresentam como soluções computacionais incompletas que, estendidas, permitem produzir diferentes artefatos de *software*. Ao oferecer a possibilidade do reuso de código e projeto permitem a redução de tempo e esforço para a obtenção de um sistema, propiciando, assim a geração de aplicações caracterizadas pela qualidade, baixo custo e rapidez no desenvolvimento. Em aplicações LBS existe um conjunto de funcionalidades ou serviços que podem ser reutilizados. O FRAGIL, através de sua modularidade e estensibilidade, apresenta-se como uma alternativa para ajudar os desenvolvedores de *software* a criar aplicações LBS fornecendo um suporte ao gerenciamento da localização dos objetos móveis cadastrados no sistema.

A utilização do *framework* proposto para construir uma aplicação LBS propiciou aos desenvolvedores uma grande abstração do processo de gerenciamento da informação de localização, fazendo com que os esforços no processo de construção do sistema estivessem concentrados apenas nas questões relacionadas ao domínio da aplicação. Tais fatos permitiram um ganho significativo no tempo utilizado para a construção do sistema.

A utilização da arquitetura e do *framework* mostrou-se satisfatória em um cenário onde a quantidade de usuários móveis a ser monitorada não fosse muito elevada. Em aplicações que demandam a atualização da informação de localização para centenas de objetos, faz-se necessária uma abordagem distribuída dos servidores de localização. Nesse caso, poderia se pensar em uma arquitetura onde cada *LS* seria responsável por administrar os clientes que estivessem dentro da região delimitada para aquele servidor. Para tanto, um mecanismo deve existir de forma a fazer a transferência dos dados necessários entre os *LSs* ao

detectar que o usuário saiu da área de atuação de *LS* específico e entrou na região monitorada por outro servidor de localização.

A precisão da informação de localização fornecida pelos PDEs é de fundamental importância para aplicações LBS, uma vez que influencia diretamente na qualidade dos serviços oferecidos aos usuários. O FRAGIL, em sua versão atual, não leva em consideração a estimativa de erro (ou grau de precisão) inerente a cada tecnologia de localização, seria necessário incluir um modelo probabilístico de forma a fornecer informações mais precisas aos usuários ao utilizarem os serviços de consulta e eventos disponibilizados pelas APIs do *framework*.

Apesar do FRAGIL estar realizando as principais funcionalidades propostas inicialmente, durante o nosso trabalho de pesquisa e desenvolvimento foram detectados alguns aspectos que podem limitar o seu pleno funcionamento, dando margem a questões importantes para o aperfeiçoamento e extensão deste trabalho.

Como sugestões para trabalhos futuros, propomos a melhoria em alguns fatores do *framework* apresentado. Dentre esses, podemos destacar: a extensão para permitir o suporte a uma abordagem distribuída dos servidores de localização, de forma a permitir uma maior escalabilidade no gerenciamento dessa informação; incluir um modelo que trate a informação de localização como um dado probabilístico e não como uma certeza, fornecendo informações mais precisas aos usuários que se utilizam dos serviços de consulta e eventos; acrescentar um mecanismo de detecção e suporte à desconexão dos usuários móveis, uma vez que o ambiente da rede sem fio apresenta a característica de conectividade intermitente, sendo tipicamente comum perdas de conexão por curtos períodos.

Infelizmente, em razão da indisponibilidade da infra-estrutura necessária e por não ser o foco principal deste trabalho, a aplicação mostrada no quarto capítulo não atende a todos os requisitos funcionais de um sistema LBS completo, servindo apenas para demonstrar os

serviços oferecidos pelo *framework*. Um trabalho que implemente uma aplicação LBS em todos os seus aspectos, incluindo a utilização de uma rede sem fio e uma tecnologia para detecção da posição do usuário e utilize o FRAGIL seria de extrema valia.

Por fim, faz-se necessário ressaltar que apesar das diversas possibilidades interessantes de serviços que podem ser oferecidos por sistemas baseados na localização dos usuários, muitas pessoas ainda se mostram avessas com relação à utilização desses sistemas principalmente em decorrência da questão da invasão de privacidade. É evidente que a questão da privacidade está diretamente relacionada ao tipo de informação que está sendo oferecido. Por exemplo, em uma aplicação de rastreamento de frotas de veículos ou de gerenciamento de equipes em campo não seria apropriado pensar em uma possível invasão de privacidade. O importante é dar a possibilidade para quem está construindo a aplicação LBS de decidir de que forma e em que momentos a posição do usuário será conhecida. O FRAGIL adota uma abordagem onde o usuário decide, através do registro no servidor de localização, se quer ou não ser monitorado. Um trabalho interessante poderia ser desenvolvido no âmbito da privacidade da localização, oferecendo, por exemplo, meios para que o usuário possa definir com que nível de precisão quer ser encontrado. Assim, o usuário que está situado na rua da cruz, em frente ao Museu de Arte Sacra, no bairro do centro pode definir que os outros usuários saibam apenas que o mesmo está transitando pelo centro sem dar detalhes de sua exata localização.

REFERÊNCIAS

ABOWD, Gregory et al. **Cyberguide: A mobile context-aware tour guide.** Graphics, Visualization and Usability Centre, College of Computing, Georgia Institute of Technology, Atlanta,USA, 2000.

AGRE, Jonathan et al. **A Layered Architecture for Location-based Services in Wireless Ad Hoc Networks**, IEEE Aerospace Conference, Big Sky, MT, USA, March, 2002. Disponível em: http://www.flacp.fujitsulabs.com/FLA-PCRTM01-01-LSM.pdf . Acesso em: 20 out. 2005.

ANDERSEN, Kristian; CHENG, Michael; KLITGAARD, Rasmus. **Framework for building location based services**. Technical report, Aalborg Universitet, 2003.

AT&T. **Active Bat System**. Disponível em: <http://www.uk.research.att.com/ab.html> Acesso em: 14 set. 2005.

AUTOTRAC. **Produtos**. Disponível em: <http://www.autotrac.com.br/cgi-in/PageSvr.dll/Get?id_sec=5>. Acesso em: 10 jul. 2005.

BOERTIEN, Nicky; MIDDELKOOP, Eric. **Location Based Services:** Services and technologies. Telematica Institute, CMG. Mai. 2002.

BOONDAO, Roongrasamee; ESICHAIKUL, Vatcharaporn; TRIPATHI, Nitin Kumar. **A Model of Location Based Services for Crime Control**. Map Asia Conf. 2003.

CHEN, Y.; CHEN, X. Y.; RAO, F. Y.; YU, X. L.; LI ,Y.; LIU, D. **LORE: An infrastructure to support location-aware services**. IBM Journal of Research nd Development, volume 48, issue5/6, nov. 2004. pag. 601-615

CHON, H.; AGRAWAL D. **Data Management for Moving Objects**. IEEE Database Engineering Bulletin, volume 25, 2002.

CITYSYNC. Lonely Planet Publications. Lonely planet citysync, 2003. Disponível em: <http://www.citysync.com >. Acesso em: 10 mar. 2005.

CLDC. **JSR-000139 Connected Limited Device Configuration 1.1.** Disponível em: <http://jcp.org/aboutJava/communityprocess/final/jsr139/index.html>. Acesso em: 14 set. 2005.

DAVIES, Nigel; MITCHELL, Keith; CHEVERST, Keith; FRIDAY, Adrian. **Caches in the Air: Disseminating Tourist Information in the Guide System**, *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, 1999, 11-19. Disponível em: <http://www.guide.lancs.ac.uk/cachesintheair.pdf>. Acesso em: 14 set. 2005.

DESHPANDE, Nikhil; BORRIELLO, Gaetano. **Location-Aware Computing: Creating Innovative and Profitable Applications and Services**, Intel Research & Development, 25 ago. 2002.

DIBDIN, Peter. **Where are mobile location based services?** CM316 Multimedia Systems Paper, dec. 2001.

DJUKNIC, Goran; RICHTON, Robert. **Geolocation and Assisted GPS**. Computer, p123-125, feb. 2001.

FAYAD, M.; SCHMIDT, C.; JOHNSON, R. **Building Application Frameworks: Object Oriented Foundations of Framework Design**. John Wiley & Sons, New York, United States, 1999.

FCC - Federal Communications Commission. **FACT SHEET: FCC Wireless 911 requirements**. January 2001. Disponível em: <http://www.fcc.gov/911/enhanced/>. Acesso em : 10 jan. 2005.

FOWLER, Martin; SCOTT, Kendall. **UML Essencial: Um guia para a linguagem-padrão de modelagem de objetos.** 2Ed.BookMan, Porto Alegre, 2000.

HARGRAVE, Sean. **Mobile Location Services: A Report into the State of the Market**, White Paper, Cambridge Positioning Systems, 2000.

HARTER, Andy; HOPPER, Andy; STEGGLES, Pete; WARD, Andy; WEBSTER, Paul. **The Anatomy of a Context-Aware Application**. Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking, MOBICOM'99, pp. 59-68, Seattle, Washington, USA, ago. 1999.

HIGHTOWER, Jeffrey; BORRIELLO, Gaetano. **A Survey and Taxonomy of Location Systems for Ubiquitous Computing**. Computer, vol. 34, nº 8, pp. 57-66, IEEE Computer Society Press, ago. 2001.

HJELM, Johan. **Creating Location Services for the Wireless Web.** Wiley Publishing, New York, United States, 2002.

IGEB - INTERAGENCY GPS EXECUTIVE BOARD. **Global Positioning System – Standard Positioning Service – Performance Standard**. Assistant Secretary of Defense for Command, Control, Communications and Intelligence. Washington, DC, out. 2001. Disponível em: <http://www.pnt.gov/SPS-2001-final.pdf>. Acesso em: 19 set. 2005.

J2ME**. Java 2 Platform Micro Edition**, Sun Microsystems. Disponível em: <http://java.sun.com/j2me/> . Acesso em: 14 set. 2005.

JÄRVENSIVU, Riku; PITKÄNEN, Risto; MIKKONEN, Tommi. **Object-Oriented Middleware for Location-Aware Systems**. In the proceedings of ACM 2004 Symposium on Applied Computing, mar. 2004, pag. 1184-1190.

LARA, Walter. **Universal Location Framework: A New Wireless Building Block**. Intel DeveloperUPDATE Magazine, 2003.

LEONHARDI, A., and ROTHERMEL, K.: **Architecture of a Large scale Location Service**. Proc. of the 22nd Int.Conf. on Distributed Computing Systems (ICDCS), Vienna, Austria, 2002.

LEONHARDI, A.; ROTHERMEL, K. **A Comparison of Protocols for Updating Location Information.** Technical Report TR-2000-05. University of Stuttgart, 2000.

LIU, Zhe. **A Java-Based Wireless Framework for Location-Based Services Applications.** The University of Calgary, Jun. 2002. Disponível em: <http://www.geomatics.ucalgary.ca/links/GradTheses.html>. Acesso em: 15 set. 2005.

MALLICK, Martyn. **Mobile and Wireless Design Essentials.** Wiley Publishing, United States, 2003.

MAPINFO. **Mobile Location Services.** Disponível em: <www.mobileinfo.com/wireless/img/new%20mobile%20area.doc>. Acesso em: 15 jun. 2004.

MATTSSON, M. **Object-oriented Frameworks: A Survey of Methodological Issues**. Licentiate Thesis. Department of Computer Science, Lund University. Sweden. Disponível em: <http://www.ipd.bth.se/michaelm/papers/>. Acesso em: 15 jan. 2005.

MEDEIROS, Ernani. **Desenvolvendo software com UML 2.0: definitivo**. Pearson Makron Books, São Paulo, 2004.

MIDP. **JSR-000118: Mobile Information Device Profile 2.0.** Disponível em: <http://jcp.org/aboutJava/communityprocess/final/jsr118/> . Acesso em: 14 set. 2005.

MIRANDA, Rodrigo Vilar; BAPTISTA,Cláudio S.; ALMEIDA, Rodrigo R.; CATÃO, Bruno; PAZINATTO, Eder. **IGIS: um Framework para Sistemas de Informações Geográficas em N-Camadas usando um SGBD Objeto-Relacional**, SBC GeoInfo 2002, Caxambu, Brasil, dez. 2002.

MOURA, Ana Lúcia. **Bússola orientada por GPS.** Dezembro de 2004. Disponível em: <http://world.idg.com.br/adCmsDocumentShow.aspx?GUID=9B9BDC52-4449-46E8-A1F0-FA3D41B99474&ChannelID=42> Acesso em: 14 set. 2005.

NACIF, Mauro Rocha. **Computação Móvel: Serviços Baseados em Localização**. In: I Workshop sobre Geoprocessamento da UFV, Viçosa, 2002.

NORD, J.; SYNNES, K.; PARNES, P. **An Architecture for Location Aware Applications.** Proceedings of the Hawai'i International Conference on System Sciences (HICSS-35). Big Island, Hawaii, Volume 9, jan. 2002, pag.293

OGC, Open GIS Consortium Reference number 03-006r3. **OpenGIS Location Services (OpenLS): Core Services**. 16 Jan. 2004. Disponível em: <http://www.opengis.org/>. Acesso em: 21 ago. 2005.

OSTP - OFFICE OF SCIENCE AND TECHONOGY POLICY. **Statement by the President Regarding the United States Decision to Stop Degrading Global Positioning System Accuracy**. Washington, DC, maio 2000. Disponível em: <http://www.ostp.gov/html/0053_2.html>. Acesso em: 19 set. 2005.

PENG, Zhong Ren. **Internet GIS: Distributed Geographic information services for the internet and wireless networks.** John Wiley & Sons, New Jersey, mar. 2003.

RUI, José; MOREIRA, Adriano J. C.; MENESES, Filipe; COULSON, Geoff. **An Open Architecture for Developing Mobile Location-Based Applications over the Internet**. Proceedings of the Sixth IEEE Symposium on Computers and Communications (ISCC 2001), Hammamet, Tunisia. IEEE Computer Society 2001, jul. 2001, pag. 500-505

SANTOS, José Mauricio Pinheiro. **RFID - Identificação por Radiofreqüência**. Maio de 2004. Disponível em: <http://www.projetoderedes.com.br/artigos/artigo_identificacao_por_radiofrequencia.php>. Acesso em: 14 set. 2005.

SATOH, Ichiro. **A Mobile Agent-based Framework for Location-Based Services**, Proceedings of IEEE International Conference on Communications (ICC'2004), IEEE Communication Society, jun. 2004

SCHILLER, Jochen; VOISARD, Agnès. **Location-Based Services.** Elsevier, United States, 2004.

SHIODE, Narushige et al. **The Impact and penetration of Location-Based Services.** Centre for Advanced Spatial Analysis, University College London, May 2002. Disponível em: <http://eprints.ucl.ac.uk/archive/00000246/01/Paper50.pdf >. Acesso em: 06 set. 2005.

STOJANOVIC, Dragan; DJORDJEVIC**,** Slobodanka. **Internet GIS Application Framework for Location-Based Services Development**, 7th EC-GI & GIS WORKSHOP EGII, Potsdam, Germany, jun. 2001.

TRUEPOSITION. **E-112 Issues and Answers: Recommendations and Insight for the Optimal Planning and Implementation of E-112, Emergency Wireless Location For the European Union**. Disponível em: <www.trueposition.com/e112_issues_and_answers.pdf >. Acesso em: 14 set. 2005.

TSELIKAS, N.; KOUTSOLOUKAS, E.; KAPELLAKI, S.; CHANIOTAKIS, E. **An OSA/Parlay-Based Middleware Architecture for Location-Based Services**. Wireless Personal Communications. Volume 30 , Issue 2-4, set. 2004. pag. 247 – 265

VINDIGO. 2002. Disponível em: < http://www.vindigo.com >. Acesso em: 18 mai. 2005.

VISUAIDE. **A gps system for the blind and visually impaired. 2003.** Disponível em: <http://www.visuaide.com/gpssol.html>. Acesso em: 10 jun. 2005.

VIVO. Disponível em: <http://www.vivo.com.br/portal/vivo_encontra.php> Acesso em: 20 ago. 2005.

WANT, Roy; HOPPER, Andy; FALCAO, Veronica; GIBBSON, Jon. **The active badge location system.** ACM Transactions on Information Systems, 10(1):91-102, jan. 1992.

WEBRASKA. **ApontadorDuo**. Disponível em: <http://www.apontadorduo.com.br>. Acesso em : 13 jul. 2005.

ZURSTRASSEN, Leonardo. **LBS – *Location Based Services***. Disponível em: <http://www.wirelessbrasil.org/wirelessbr/colaboradores/zurstrassen/lbs_01.html >. Acesso em: 14 set. 2005.