

UNIVERSIDADE REGIONAL DE BLUMENAU
CENTRO DE CIÊNCIAS EXATAS E NATURAIS
CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

APLICAÇÃO MOBILE MARKETING COM COMUNICAÇÃO
BLUETOOTH FOCADA EM BARES E RESTAURANTES

RAFAEL FORMENTO

BLUMENAU
2009

2009/2-16

RAFAEL FORMENTO

APLICAÇÃO MOBILE MARKETING COM COMUNICAÇÃO

BLUETOOTH FOCADA EM BARES E RESTAURANTES

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciência da Computação — Bacharelado.

Prof. Paulo Fernando da Silva - Orientador

**BLUMENAU
2009**

2009/2-16

APLICAÇÃO MOBILE MARKETING COM COMUNICAÇÃO BLUETOOTH FOCADA EM BARES E RESTAURANTES

Por

RAFAEL FORMENTO

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

Presidente: _____
Prof. Paulo Fernando da Silva, Mestre – Orientador, FURB

Membro: _____
Prof. Francisco Adell Péricas, Mestre – FURB

Membro: _____
Prof. Sérgio Stringari, Mestre – FURB

Blumenau, 17 de dezembro de 2009

Dedico este trabalho a todos os amigos, meus pais e minha noiva, que acreditaram e incentivaram para realização deste

AGRADECIMENTOS

A Deus, pelo seu imenso amor e graça.

À minha família, que soube respeitar meu tempo e sempre me deu força pra não desistir.

A minha noiva Ana, que além de sempre estar ao meu lado incentivando e apoiando, me ajudou com idéias e teorias em relação a um mundo que está a nossa volta, nos influencia e não percebemos que está ali. O mundo do marketing.

Aos meus amigos, por sempre me ajudarem e apoiarem nos momentos difíceis, principalmente André Luiz Lehmann, que foi praticamente um consultor em Java.

Ao meu orientador, Paulo Fernando da Silva, por ter acreditado na conclusão deste trabalho e por me mostrar a luz quando eu estava completamente perdido.

Um raciocínio lógico leva você de A a B. A imaginação leva você a qualquer lugar que você quiser.

Albert Einstein

RESUMO

Este trabalho apresenta o desenvolvimento de uma aplicação no conceito de *Mobile Marketing*, ou seja, marketing através de dispositivos móveis, possibilitando a interação entre um computador e um PDA. Esta interação é feita através da tecnologia de rede *Bluetooth*. A implementação permite auxiliar um estabelecimento de venda (uma lanchonete, por exemplo) com ações de marketing, como venda de produtos personalizada, promoções e enquetes. A utilização da plataforma Java, da tecnologia J2ME e J2SE junto com o *framework* Marge foram componentes necessários para a implementação desta ferramenta.

Palavras-chave: Mobile marketing. Bluetooth. Bluecove. J2ME. Marge.

ABSTRACT

This paper presents the development of an application in the concept of Mobile Marketing, or marketing via mobile devices, enabling the interaction between a computer and a PDA. This interaction is done through the Bluetooth networking technology. The implementation allows an auxiliary outlet (a cafeteria, for example) with marketing actions, such as selling customized products, promotions and surveys. The use of the Java platform, the J2ME and J2SE technology along with the Marge framework components were required to implement this tool.

Key-words: Mobile marketing. Bluetooth. Bluecove. J2ME. Marge.

LISTA DE ILUSTRAÇÕES

Figura 1 – Representação de um cenário de computação móvel.....	16
Quadro 1 - Características principais dos canais físicos <i>Bluetooth</i>	18
Figura 2 - Intervalo de tempo no salto de frequência.....	19
Figura 3 - Sequência de comandos de reconhecimento.....	20
Figura 4 - Representação de uma rede <i>piconet</i>	20
Figura 5 - Representação de uma rede <i>scatternet</i>	21
Figura 6 – Camadas da arquitetura J2ME	22
Figura 7 - Visão geral dos componentes da tecnologia Java ME e como se relaciona com outras tecnologias Java.	23
Figura 8 - Arquitetura <i>Bluetooth</i> e onde o Marge se encaixa.....	26
Figura 9 - Diagrama da arquitetura Bluecove na comunicação com J2SE à J2ME.....	27
Figura 10 – Tela para informar itens e tela de compra efetuada	29
Figura 11 – Disposição dos personagens no cenário do jogo.....	30
Quadro 2 - Requisitos não funcionais	32
Quadro 3 - Requisitos funcionais	33
Figura 12 - Diagrama de casos de uso do módulo servidor executados pelo usuário	34
Figura 13 - Diagrama de casos de uso do módulo servidor executados pela aplicação	34
Quadro 4 - Detalhamento do caso de uso Enviar dados do pedido.....	35
Quadro 5 - Detalhamento do caso de uso Enviar dados das promoções.....	35
Quadro 6 - Detalhamento do caso de uso Enviar dados das enquetes.....	36
Figura 14 - Diagrama de casos de uso da aplicação módulo cliente.....	36
Quadro 7- Detalhamento do caso de uso Enviar dados do pedido efetuado.....	37
Quadro 8 - Definições dos pacotes de manipulação de dados e de interface com usuário	38
Figura 15 - Pacote com estrutura da comunicação <i>Bluetooth</i>	39
Figura 16 - Classes do pacote que trata as mensagens recebidas via <i>Bluetooth</i>	40
Figura 17 - Pacote contendo a estrutura das telas e da comunicação	41
Figura 18 - Pacote contendo as classes manipulam as mensagens.....	42
Figura 19 - Diagrama de sequência para enviar pedido	43
Figura 20 - Diagrama de sequência para cliente receber promoções	44
Figura 21 - Diagrama de sequência para cliente receber enquetes.....	44
Quadro 9 - String compactada recebida de um cliente.....	46

Quadro 10 - String enviada do servidor para um cliente.....	46
Quadro 11 - String enviada por um cliente.....	46
Quadro 12 - Troca de mensagens no serviço de Promoção e pesquisa.....	47
Quadro 13 - Inicialização do servidor na classe <code>RFCOMMServer</code>	47
Quadro 14 - Método implementado para envio de mensagens.....	48
Quadro 15 - Método invocado ao encontrar algum dispositivo com <i>Bluetooth</i> ativo.....	48
Quadro 16 - Método implementado para executar após a conexão estabelecida.....	49
Quadro 17 - Método que monitora o recebimento de mensagens vindas do servidor.....	49
Quadro 18 - Método que chama telas conforme requisição.....	50
Quadro 19 - Possíveis utilizações da implementação.....	51
Figura 22 - Tela do cadastro de cliente.....	52
Figura 23 - Tela de cadastro de cardápio.....	53
Figura 24 - Tela de escolha do serviço no cliente.....	54
Figura 25 - Três passos para efetuar um pedido no módulo cliente.....	54
Figura 26 - Tela com informação dos pedidos recebidos.....	55
Figura 27 - Tela de cadastro de campanhas.....	56
Figura 28 - Telas de campanhas no módulo cliente.....	57
Figura 29 - Tela de cadastro de enquetes.....	58
Figura 30 - Telas de enquetes no módulo cliente.....	58
Figura 31 - Tela fluxo de dados de cada usuário.....	59
Quadro 20 – Características da aplicação e trabalhos correlatos.....	60

LISTA DE SIGLAS

API – *Application Programming Interface*

CDC – *Connected Device Configuration*

CLDC – *Connected Limited Device Configuration*

JSR – *Java Specification Requests*

IEEE - *Institute of Electrical and Electronics Engineers*

KVM – *Kilobyte Virtual Machine*

MAC – *Media Access Control*

MIDP - *Mobile Information Device Profile*

PDA - *Personal digital assistants*

PBP - *Personal Basis Profile*

RF – Requisito Funcional

RNF – Requisito Não Funcional

UML – *Unified Modeling Language*

XML - *eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO.....	13
1.1 OBJETIVOS DO TRABALHO	14
1.2 ESTRUTURA DO TRABALHO	14
2 FUNDAMENTAÇÃO TEÓRICA.....	15
2.1 MOBILIDADE.....	15
2.1.1 Dispositivos móveis	16
2.1.2 <i>Bluetooth</i>	17
2.1.2.1 História	17
2.1.2.2 Arquitetura	18
2.1.2.3 Conexões.....	19
2.2 BIBLIOTECAS ENVOLVIDAS	21
2.2.1 J2ME	21
2.2.1.1 Configuração CLDC e CDC.....	23
2.2.1.2 Perfil MIDP e PBP.....	24
2.2.2 Marge	25
2.2.2.1 Arquitetura	25
2.2.3 Bluecove.....	27
2.3 MOBILE MARKETING.....	28
2.3.1 <i>Bluetooth</i> Marketing.....	28
2.4 TRABALHO CORRELATOS	29
3 DESENVOLVIMENTO.....	32
3.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO.....	32
3.2 ESPECIFICAÇÃO	33
3.2.1 Diagramas de casos de uso.....	33
3.2.1.1 Módulo servidor.....	34
3.2.1.2 Módulo cliente	36
3.2.2 Diagrama de classes	37
3.2.2.1 Módulo servidor.....	37
3.2.2.2 Módulo cliente	41
3.2.3 Diagrama de sequência	42
3.3 IMPLEMENTAÇÃO	45

3.3.1 Técnicas e ferramentas utilizadas.....	45
3.3.2 Desenvolvimento da aplicação.....	45
3.3.2.1 Módulo servidor.....	45
3.3.2.2 Módulo cliente.....	48
3.3.3 Operacionalidade da implementação.....	50
3.4 RESULTADOS E DISCUSSÃO.....	59
4 CONCLUSÕES.....	61
4.1 EXTENSÕES.....	62
REFERÊNCIAS BIBLIOGRÁFICAS.....	63

1 INTRODUÇÃO

Segundo a Agência Brasil (2009), o número de aquisições de celulares chegou a 1,3 milhões no mês de janeiro de 2009, totalizando 151,9 milhões de assinantes de telefonia móvel no Brasil. Percebe-se com isso, que cada vez mais aumenta o número de pessoas que buscam as facilidades que a telefonia móvel pode oferecer no dia-a-dia.

Com a evolução da tecnologia de redes e serviços, assim como de novos aplicativos, a possibilidade de trazer para um celular tarefas como conectar-se a internet, controlar contas bancárias ou até mesmo divertir-se, relacionar-se, procurar um bom lugar para jantar, ouvir música, jogar e gravar vídeos e fotos ficou muito mais acessível (ROMÁN; GONZÁLEZ-MESONES; MARINAS, 2007).

Para a comunicação entre celulares e outros dispositivos, existe o *Bluetooth*. Segundo Henrique (2007), *Bluetooth* é um meio de comunicação de dados de curta distância, aproximadamente 10 metros, que tem a mesma frequência do *Wi-Fi*¹ e a capacidade de interagir com diferentes dispositivos. Há alguns anos foram feitas as primeiras campanhas de comunicação móvel, através de *Bluetooth*, utilizando apenas transferência de imagens para propaganda. Essa é uma opção muito viável para ser usada como *Mobile Marketing*.

Existem formas de produzir publicidade, relacionamento com clientes, convites para eventos usando a mídia do celular. Neste contexto, o celular é o canal ou a mídia, enquanto as estratégias e a forma ficam para o *Mobile Marketing* (LANGE; SHROEDER, 2008).

Tendo em vista a demanda do mercado, com cada vez mais pessoas interessando-se pelo desenvolvimento da área de aplicativos *mobile* e com todos os avanços tecnológicos nesta área, foi implementado uma ferramenta para viabilizar o uso de *Mobile Marketing*, enviando e recebendo alguns tipos de publicidade, como fotos, vídeos e questionários. Para o envio e recebimento desta publicidade, é usada a comunicação por meio de *Bluetooth*. Com o intuito de facilitar a implementação e o entendimento da ferramenta, é usado o *framework* Marge e o protocolo de comunicação Bluecove. Com o uso destas duas ferramentas será possível enviar mensagens de um servidor à um celular através do meio *Bluetooth*.

¹ Comumente o termo *Wi-Fi* é entendido como uma tecnologia de interconexão entre dispositivos sem fios, usando o protocolo IEEE 802.11.

1.1 OBJETIVOS DO TRABALHO

O objetivo deste trabalho é o desenvolvimento de uma aplicação que torne possível a interação entre um PDA e um computador por meio *Bluetooth*.

São objetivos específicos do trabalho:

- a) permitir a comunicação entre uma aplicação servidora e um dispositivo móvel através de *Bluetooth*, usando o protocolo *Bluecove*;
- b) visualizar em uma aplicação servidora os clientes com *Bluetooth* ativo em seu dispositivo e com permissão de receber publicidade;
- c) oferecer uma aplicação *mobile* para interagir com a aplicação servidora usando J2ME para o aplicativo móvel e J2SE à aplicação servidora;
- d) permitir que a aplicação servidora armazene os clientes que desejam continuar recebendo informações.

1.2 ESTRUTURA DO TRABALHO

O presente trabalho está organizado em três capítulos intitulados respectivamente como: fundamentação teórica, desenvolvimento e conclusões.

O capítulo 2 apresenta os aspectos teóricos estudados para o desenvolvimento do trabalho. São abordados temas como mobilidade, *Bluetooth*, bibliotecas, *mobile* marketing, que foram utilizados neste trabalho. Também são relacionados alguns trabalhos correlatos.

No capítulo 3 é descrito como foi realizado o desenvolvimento deste trabalho, detalhando os requisitos do protótipo, a especificação e a implementação. São apresentados os resultados encontrados com a finalização do trabalho.

Por fim, o capítulo 4 traz conclusões deste trabalho, bem como alguns aspectos que ficaram em aberto, servindo de sugestões para futuras extensões.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta inicialmente conceitos sobre mobilidade. Em seguida aborda conceitos de dispositivos móveis e o meio de comunicação sem fio *Bluetooth*. Na sequência são relacionadas as bibliotecas envolvidas na ferramenta, como J2ME e o framework Marge, e ainda definições de *Mobile Marketing*. Por fim, são apresentados trabalhos correlatos que auxiliaram no entendimento e no conhecimento de aplicativos móveis.

2.1 MOBILIDADE

Mobilidade, segundo Lee, Schneider e Schell (2005), é a capacidade de poder deslocar-se facilmente. Na visão computacional, a mobilidade pode ser atribuída ao uso de dispositivos móveis funcionais e com capacidade de conectar-se, obter dados e fornecê-los a outros usuários, aplicações e sistemas.

Para ser considerado móvel, um dispositivo deve possuir determinadas características, como a portabilidade, usabilidade, funcionalidade e conectividade:

- a) portabilidade: quando um dispositivo pode ser facilmente transportado. Atualmente, um dispositivo pode ser considerado portátil quando tem a capacidade de ser transportado na palma da mão;
- b) usabilidade: quando um dispositivo pode ser usado em qualquer ambiente e por qualquer pessoa;
- c) funcionalidade: quando um dispositivo pode ser usado para várias aplicações. Atualmente, dispositivos têm várias aplicações móveis rodando neles. Em geral, essas aplicações se encaixam em duas categorias: dependentes, quando necessitam conectar-se a outros usuários ou aplicações, como por exemplo calendários, notícias ou GPS; independentes, ou seja, sem a necessidade de conectar-se a outros usuários ou aplicações, por exemplo relógio, jogos ou calculadora;
- d) conectividade: é a capacidade de um dispositivo conectar-se com outros dispositivos ou usuários. Uma conexão não necessariamente se faz com conexões sem fio. Qualquer interatividade com outro usuário ou dispositivo através de qualquer meio, pode ser considerado uma conexão.

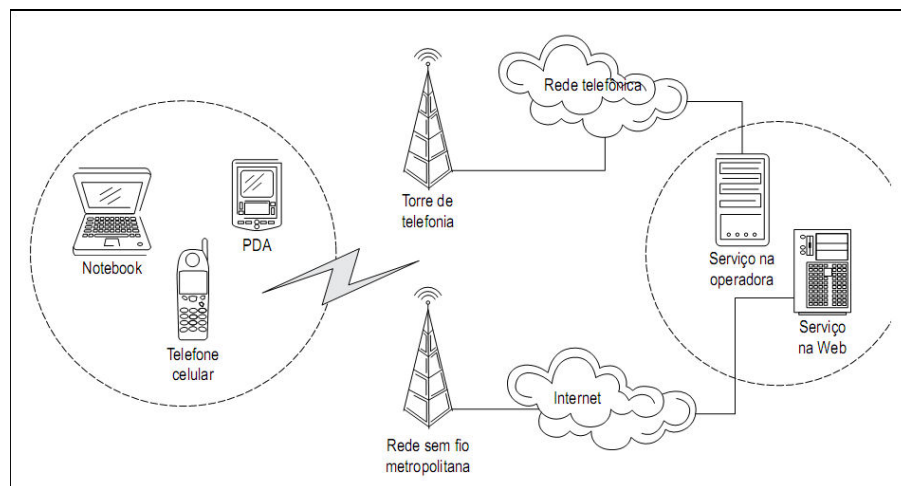
2.1.1 Dispositivos móveis

Segundo Johnson (2007), existem diversos grupos de dispositivos que podem ser considerados como móveis. Em um nível mais elevado de poder de processamento, estão os laptops, que são computadores portáteis com a mesma capacidade dos computadores desktop.

Em seguida vem os *Personal Digital Assistants* (PDAs), que têm uma tela pequena, mas um alto poder de processamento se comparado a um celular. Um PDA tem suporte a arquivos multimídia, aplicativos desenvolvidos com linguagens de programação de alto nível, acesso a rede, etc.

O terceiro grupo são os telefones celulares. Os primeiros celulares possuíam telas pequenas, baixo poder de processamento e pouca memória. Com sua evolução vieram aparelhos mais robustos e poderosos recursos funcionais. Alguns, com o alto poder de processamento, memória e funcionalidades, foram chamados de *Smartphones*, pois são celulares, mas tem alguns recursos dos PDAs.

Todos estes dispositivos têm características em comum, podem compartilhar a mesma aplicação. A Figura 1 apresenta um cenário onde dispositivos se conectam sem fio a diversos serviços. Através de uma torre de telefonia, o dispositivo acessa os serviços da operadora, ou uma torre de internet, onde o dispositivo acessa os serviços na Web.



Fonte: adaptado de Johnson (2007).

Figura 1 – Representação de um cenário de computação móvel

2.1.2 *Bluetooth*

Conforme Alecrim (2008), *Bluetooth* é uma tecnologia que permite a comunicação barata, rápida e segura entre dispositivos distintos como *smartphones*, telefones celulares, mouses, teclados, fones de ouvido, impressoras e outros dispositivos, utilizando ondas de rádio no lugar de cabos. Isso possibilita a troca de informações entre dois dispositivos ou mais, estando apenas um perto do outro.

Segundo Rufino (2007), *Bluetooth* é a tecnologia que mais se aproxima da Wi-Fi (usa a mesma frequência: 2,4GHz) e está disponível em vários equipamentos. Esta tecnologia ainda permite recursos comparáveis aos de redes convencionais. Foi criada para ser uma rede de baixo custo, baixa complexidade e pouca potência para cobrir curtas distâncias. A proposta deste protocolo (definido pelo *Institute of Electrical and Electronics Engineers 802.15 (IEEE 802.15)*) foi de uso para aplicativos móveis e transmissão de dados e voz (celular, palm, etc.), mas também para outros dispositivos como fone de ouvido ou até mesmo impressoras.

O IEEE 802.15, segundo Santana (2009), é um conjunto de especificações que define o protocolo e seu comportamento em uma transmissão entre rádios. Estas especificações tornam possíveis aos dispositivos de monitoração ou controle, características como links confiáveis, tempo de latência da mensagem e a possibilidade de implementação em dispositivos dotados de poucos recursos computacionais. E também define o comportamento das camadas Física, e de Controle de Acesso ao Meio (MAC).

2.1.2.1 História

Em 1994, a Ericson Mobile Communications reconheceu e autorizou um estudo para uma alternativa aos cabos usados para conectar celulares e outros dispositivos. A pesquisa focalizou uma interface de baixo custo e baixo consumo de energia entre os telefones móveis e seus acessórios.

Para acelerar o desenvolvimento da tecnologia, a Ericson resolveu se juntar a outras empresas (IBM, Intel, Nokia e Toshiba). Assim, em 1998 foi anunciado uma nova tecnologia de comunicação chamada *Bluetooth*, e também o grupo Special Interest Group (SIG). O SIG foi encarregado de monitorar o desenvolvimento das tecnologias de rádio de curta distância e criar um padrão global aberto à comunicação *Bluetooth* (MILLER, 2001).

Segundo Alecrim (2008), a nomenclatura *Bluetooth* foi definida em memória a um rei dinamarquês chamado Harald Bluetooth (Haroldo Dente-Azul). O vínculo com a tecnologia foi o fato de Harald ter unificado a Dinamarca, da mesma forma com que a tecnologia propõem a unificação de vários dispositivos, como celulares, computadores, impressoras, etc. O logotipo *Bluetooth* é a junção de dois símbolos nórdicos que correspondem às iniciais de Harald.

2.1.2.2 Arquitetura

Para Alencar (2004), a interface de comunicação *Bluetooth* foi desenvolvida com o objetivo de integrar um dispositivo de rádio a um computador. *Bluetooth* pode ser uma alternativa à interface de comunicação unidirecional com radiação infravermelha, oferecida pela *Infrared Data Association* (IrDA). O infravermelho limita a transmissão a um metro de distância e com ausência de qualquer obstrução.

Bluetooth pode ser considerada uma tecnologia a ser usada no mundo todo. Portanto é necessário o uso de uma frequência aberta, que seja padrão em todo lugar. Para isso, é usado a faixa *Industrial, Scientific, Medical* (ISM), que opera na frequência 2,45GHz. O Quadro 1 apresenta características principais dos canais físicos que permitem a transmissão de dados entre dois dispositivos *Bluetooth*.

Parâmetro	Informação / Valor
Antena	Omnidirecional
Faixa de Frequências	2,4 GHz a 2,483 GHz
Modulação	GFSK (<i>Gaussian Frequency Shift Keying</i>)
Taxa de símbolos	1 Mega Símbolo/seg. (1 MS/s)
Nº de Canais	79
Banda do Canal	1 MHz
Banda de Guarda	Inferior 2 MHz, superior 3,5 MHz
Potência de transmissão	Classe 1: 1 (0 dBm) a 100 mW (20 dBm) Classe 2: 0,25 (-6 dBm) a 2,5 mW (4 dBm) - nominal = 1 mW Classe 3: <= 1 mW (0 dBm)
Espalhamento Espectral	Salto de frequência (<i>Frequency-Hopping</i>) a cada 625 micro segundo (μ s)

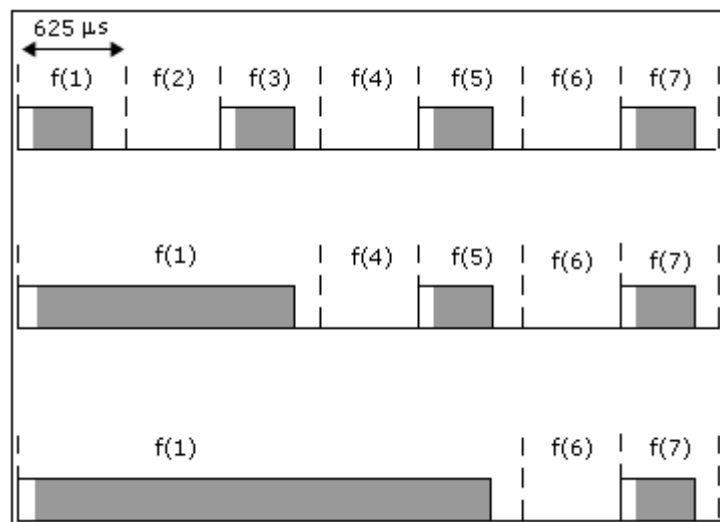
Fonte: Tude (2008).

Quadro 1 - Características principais dos canais físicos *Bluetooth*

Por ser uma frequência aberta, é necessário que não tenha interferência no sinal. Para isso, há o esquema de comunicação *Frequency Hopping - Code-Division Multiple Access* (FH-CDMA), utilizado pelo *Bluetooth*, que permite tal proteção fazendo com que a frequência se divida em vários canais. A troca de canais é feita de forma muito rápida pelo dispositivo que estabelece a conexão, e essa troca é chamada de salto de frequência. Com isto,

a largura da frequência de banda se torna muito pequena, fazendo com que as chances de interferência seja mínima. Pode ser utilizada até 79 frequências (ou 23, dependendo do país), espaçada uma da outra por 1MHz, dentro da faixa ISM (Alecrim, 2008). Na sincronização entre dois dispositivos, o dispositivo que envia a solicitação de conexão, envia também um pacote de sincronização contendo a sequência das frequências que serão alternadas (MILLER, 2001).

O intervalo de tempo no salto de frequência é de 0,625 useg. (microsegundo). Segundo Tude (2004b) o intervalo dessa transmissão em uma frequência é chamada de *slot*, ou seja um pacote de dados é transmitido em cada *slot* de tempo. O pacote pode ser estendido para ocupar 3 ou 5 *slots* de modo a aumentar a taxa de dados transmitida como apresentado na Figura 2.



Fonte: Tude (2004).

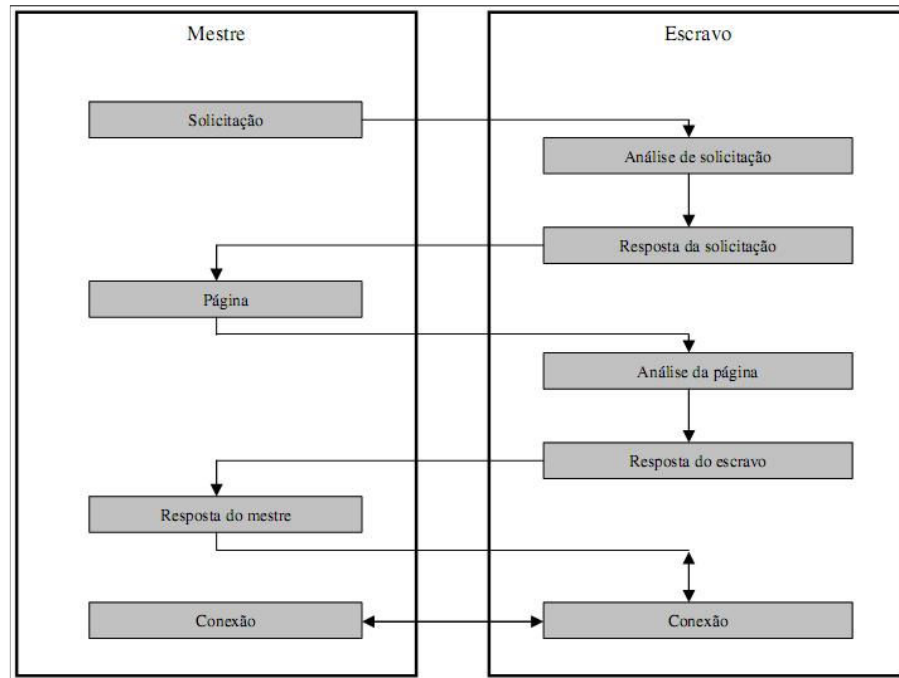
Figura 2 - Intervalo de tempo no salto de frequência

2.1.2.3 Conexões

Segundo Miller (2001) o procedimento de iniciar uma conexão *Bluetooth* pode emitir dois tipos diferentes de comandos:

- comando de solicitação: é emitido quando o número de identificação ou endereço não são conhecidos;
- comando página: é emitido quando o dispositivo já é conhecido. Assim, é enviado um comando que serve para despertar a outra unidade e estabelecer uma conexão.

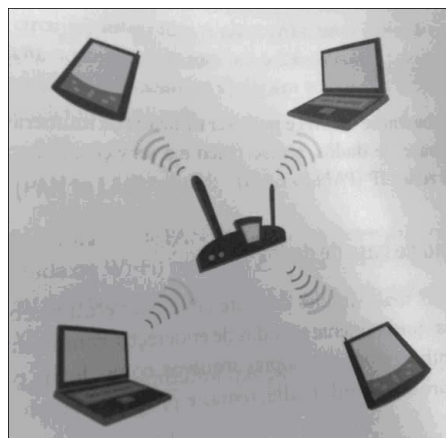
A Figura 3 mostra a sequência de mensagens que um mestre envia à um escravo, onde são enviadas as solicitações e páginas pelo mestre e analisadas pelo escravo, para que uma conexão *Bluetooth* possa ser estabelecida.



Fonte: Miller (2001).

Figura 3 - Sequência de comandos de reconhecimento

Ainda conforme Miller (2001), a partir do momento em que uma conexão é estabelecida, pode ser considerada uma rede chamada *piconet*. Cada *piconet* pode ser composta por no máximo oito dispositivos, sendo que um deles deve ser necessariamente um mestre e os outros escravos. Cada *piconet* contém uma sequência de salto de frequência distinto, possibilitando assim, cada dispositivo estar em *piconet* diferente e também haver várias *piconets* em um mesmo ambiente físico, uma vez que cada *piconet* tem uma identidade baseada em canais de salto de frequência diferentes. Um exemplo de *piconet* pode ser verificada na Figura 4.

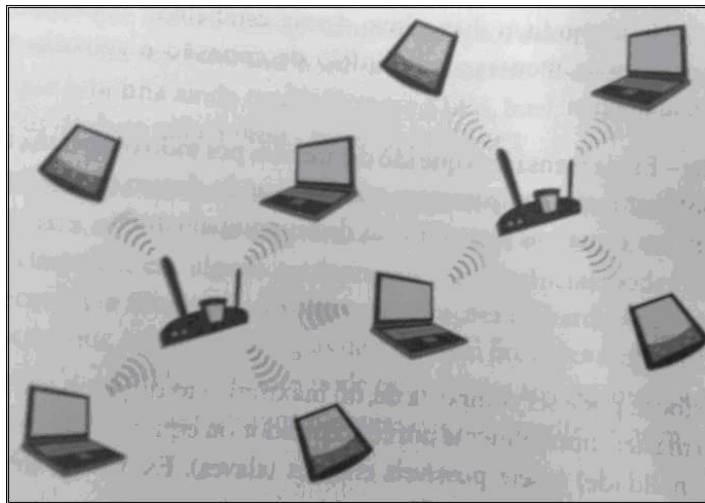


Fonte: Rufino (2007).

Figura 4 - Representação de uma rede *piconet*

Para expandir uma rede *Bluetooth*, é necessário fazer conexões entre *piconets*, denominando assim uma *scatternet*. Como o número máximo do salto de frequência é de 79

saltos e cada rede *piconet* contém até oito dispositivos, uma *scatternet* pode conter no máximo dez redes *piconet* interligadas. A Figura 5 ilustra uma *scatternet*.



Fonte: Rufino (2007).

Figura 5 - Representação de uma rede *scatternet*

2.2 BIBLIOTECAS ENVOLVIDAS

Para o desenvolvimento de uma aplicação *mobile*, são necessários recursos, que serão apresentados nas seções seguintes, tais como a biblioteca *Java Micro Edition* (Java ME), que segundo Johnson (2007) é uma versão especial da plataforma Java com a intenção de dar suporte a dispositivos portáteis. Bluecove (implementação da *Application Programming Interface* (API²) JSR-82³), que é um protocolo para comunicação *Bluetooth*. E por fim, um framework chamado Marge, que auxilia na implementação de uma aplicação com comunicação *Bluetooth*.

2.2.1 J2ME

Em 1998, a Sun lançou uma tecnologia chamada *Personal Java* (pJava), com o

² API: é um conjunto de rotinas e padrões estabelecidos por um software para a utilização das suas funcionalidades por programas aplicativos que não querem envolver-se em detalhes da implementação do software, mas apenas usar seus serviços.

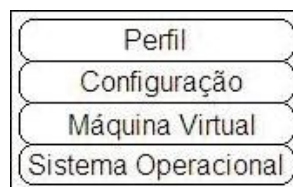
³ JSR-82: é um pacote opcional que permite a comunicação utilizando a tecnologia sem fio Bluetooth.

propósito do desenvolvimento de aplicações para dispositivos móveis. Segundo Johnson (2007), essa tecnologia funcionou bem para alguns dispositivos. Porém, para alguns aparelhos, como celulares, essa tecnologia não adaptou-se bem. Então, em 1999 a Sun lançou a plataforma J2ME, para a possibilidade do desenvolvimento de aplicações para esses aparelhos.

O pacote de desenvolvimento J2ME, por ser destinado à aplicações em dispositivos com poucos recursos de processamento e memória, e também pela máquina virtual Java dos dispositivos móveis ser bastante reduzida, tem poucos recursos da linguagem.

Para Muchow (2004), J2ME é destinado à dispositivos com baixo poder de processamento e poucos recursos de memória (como celulares ou pagers). Estes dispositivos não têm capacidade de navegar na internet ou instalar software, além daqueles que são instalados na sua fabricação. Com a implementação J2ME estes dispositivos acabam tornando-se dinâmicos em relação aos aplicativos instalados.

Johnson (2007) afirma que a plataforma J2ME tornou independente a implementação do tipo do dispositivo móvel. Esta independência é possível quando os dispositivos disponibilizam as mesmas configurações e perfis. J2ME foi dividida em camadas para permitir a utilização da mesma plataforma em diferentes dispositivos. Na Figura 6 é possível ver as camadas separadas.

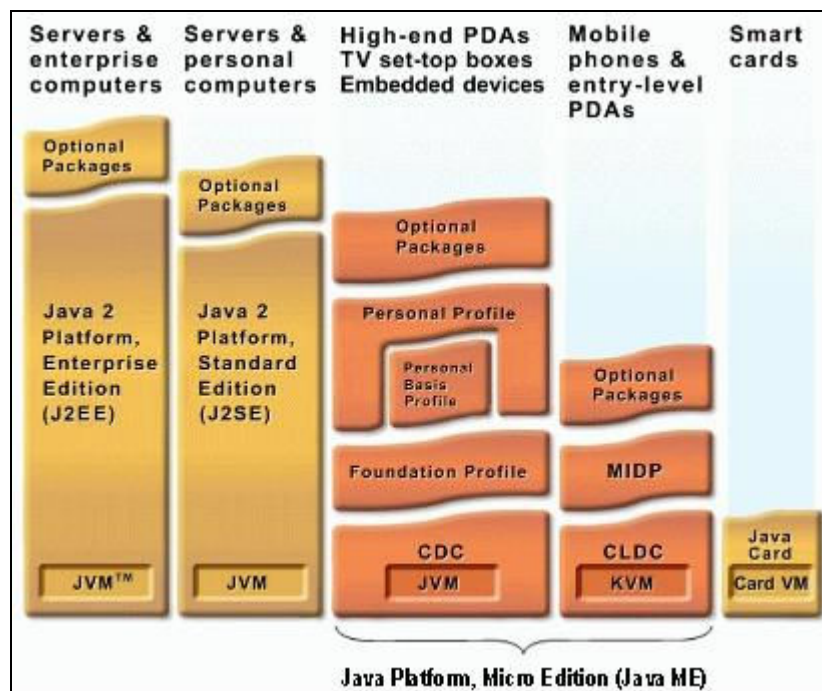


Fonte: Johnson (2007).

Figura 6 – Camadas da arquitetura J2ME

A camada máquina virtual é desenvolvida conforme a configuração escolhida. Esta escolha é feita dependendo do dispositivo. Acima desta camada, observa-se a camada de configuração, onde são disponibilizadas funcionalidades para cada categoria de dispositivos. Cada configuração oferece API's, bibliotecas e diversas classes aos dispositivos móveis. Por fim, a camada de perfil, onde são disponibilizadas API's mais específicas.

A Figura 7, disponibilizada pela Sun (2009), apresenta uma visão geral da tecnologia Java, na qual é possível visualizar o J2ME. A primeira camada, no caso dos celulares (coluna *Mobile phones & entry-level PDAs*), é a KVM, seguida pela camada de configuração (CLDC ou CDC). Logo acima visualiza-se a camada de perfil (MIDP ou PBP). Nesta figura a diferença de estrutura entre as tecnologias Java J2EE, J2SE, J2ME e a Java Card. As duas últimas com um poder visivelmente menor de recursos, permitindo assim o uso em celulares, com um poder de processamento menor comparado à um computador.



Fonte: Sun (2009).

Figura 7 - Visão geral dos componentes da tecnologia Java ME e como se relaciona com outras tecnologias Java.

2.2.1.1 Configuração CLDC e CDC

Para Muchow (2004), a camada de configuração define uma plataforma Java para uma ampla variedade de dispositivos, e está vinculada diretamente a máquina virtual. São definidos os recursos e bibliotecas básicas da linguagem Java para cada configuração em particular. A arquitetura J2ME provê dois tipos e configurações: *Connected Limited Device Configuration* (CLDC) e *Connected Device Configuration* (CDC).

A configuração CLDC é voltada à dispositivos com capacidade inferior de processamento e memória. Segundo Johnson (2007), um dispositivo com suporte a CLDC

tem muitas restrições de memória, processamento e capacidade gráfica. Esta configuração traz algumas limitações, como a de não ter suporte a operações com ponto flutuante. É usada uma máquina virtual chamada *Kilobyte Virtual Machine (KVM)*, implementada para executar programas em dispositivos com pouca capacidade computacional.

Para Mattos (2005), a configuração CLDC é voltada para o desenvolvimento em dispositivos como celulares, pagers, PDA's e outros dispositivos semelhantes, cuja limitação refere-se aos recursos de memória, processamento e, principalmente, da conectividade com redes e Internet.

A configuração CDC é voltada para dispositivos com uma maior capacidade computacional, permitindo a utilização de todos os recursos da linguagem Java, diferentemente do que ocorre na CLDC. A máquina virtual para esta configuração é a *HotSpot Implementation* (JOHNSON, 2007).

2.2.1.2 Perfil MIDP e PBP

Um perfil é uma extensão de uma determinada configuração. Cada perfil inclui novas bibliotecas relacionadas ao tipo do dispositivo, isso é, as API's usadas serão voltadas a certo grupo de dispositivos. Caso a aplicação contenha apenas os recursos padrões da plataforma, a portabilidade do *software* é garantida. Atualmente temos definidos alguns perfis, como o *Mobile Information Device Profile (MIDP)* e o *Personal Basis Profile (PBP)* (JOHNSON, 2007).

De acordo com Muchow (2004), MIDP (perfil de dispositivo de informação móvel) é o perfil utilizado para a configuração CLDC, e define API's para componentes, entrada e tratamento de eventos de interface com o usuário, armazenamento persistente, sendo sempre relevante as limitações de tela e memória dos dispositivos móveis.

O PBP é voltado para dispositivos com *interface* gráfica e não tem compatibilidade total com *Abstract Window Toolkit (AWT)*, como alguns dispositivos eletrônicos (menu de um computador de bordo de um automóvel, por exemplo) (JOHNSON, 2007).

2.2.2 Marge

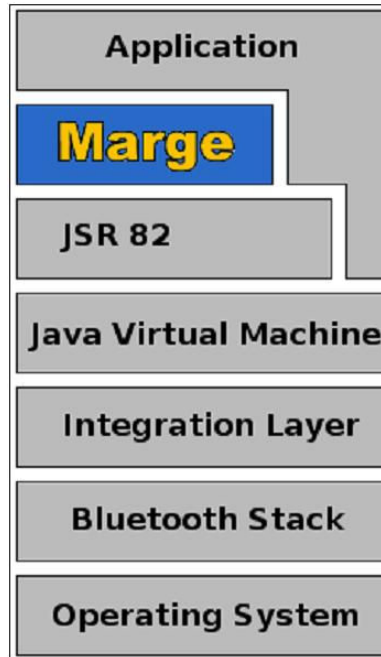
Marge⁴ é um Trabalho de conclusão de curso criado na Universidade Federal de Santa Catarina (UFSC) pelo aluno Bruno Cavaler Ghisi em 2007. Segundo Torri e Ghisi (2008), Marge é um projeto licenciado pela *Lasser General Public Licence* (LGPL), sendo mantido por colaboradores em várias partes do mundo. O projeto foi criado para ajudar no desenvolvimento de várias aplicações que necessitam do uso do *Bluetooth*, como por exemplo: jogos, controles remotos, aplicações de marketing, entre outras.

Para Ghisi (2006), autor do trabalho “Marge: Framework para desenvolvimento de aplicações em Java que façam uso da tecnologia *Bluetooth*”, o principal objetivo foi desenvolver um *framework* com capacidade de abstrair do programador a parte responsável pela comunicação *Bluetooth* em Java, definida pela JSR 82. No trabalho foram feitos estudos da plataforma J2ME e J2SE, em seguida a implementação do *framework* e no final alguns aplicativos teste com intuito de validar o *framework*.

2.2.2.1 Arquitetura

Marge é um *framework* responsável por abstrair o uso do pacote `javax.bluetooth` da JSR 82 de forma a deixá-lo mais apresentável (TORRI; GHISI, 2008). A Figura 8 apresenta a arquitetura para desenvolvimento de aplicações com *Bluetooth*. Na primeira camada – de baixo para cima - encontra-se o sistema operacional, seguido de uma pilha de protocolos *Bluetooth*, e de uma camada de integração entre a máquina virtual Java e a chamada para a pilha. Logo depois, acima da máquina virtual Java, está a JSR 82, a API que especifica o *Bluetooth* sobre a qual o Marge foi desenvolvido. Acima do Marge, encontram-se as aplicações (GHISI, 2006).

⁴ Marge: o nome tem origem na personagem do desenho animado do original em inglês ‘The Simpsons’ © 20th Century Fox Television, criado por Matt Groening,, chamada Marge. Ela é a mãe da família e possui um cabelo azul encaracolado, que é utilizado no projeto para fazer uma analogia a uma rede Bluetooth.



Fonte: Ghisi (2006).

Figura 8 - Arquitetura *Bluetooth* e onde o Marge se encaixa

O segundo Torri e Ghisi (2008), Marge é dividido em vinte e quatro classes e seis pacotes, e cada pacote é destinado a uma função, podendo ser utilizados de forma independente. Os pacotes utilizados são:

- a) `net.java.dev.marge.inquiry`: classes para a busca de dispositivos e serviços;
- b) `net.java.dev.marge.enty` – classes para abstrair o processo de comunicação, utilizando o conceito de dispositivos clientes e servidores;
- c) `net.java.dev.marge.enty.config` – configurações para esses dispositivos, que são utilizados pelas fábricas⁵;
- d) `net.java.dev.marge.factory` – fábricas que criam esses dispositivos segundo um protocolo de comunicação, por meio das configurações fornecidas;
- e) `net.java.dev.marge.communication` – classes utilizadas nas trocas de mensagens;
- f) `net.java.dev.marge.communication.layer` – camadas que podem ser utilizadas para processarem a troca de mensagens.

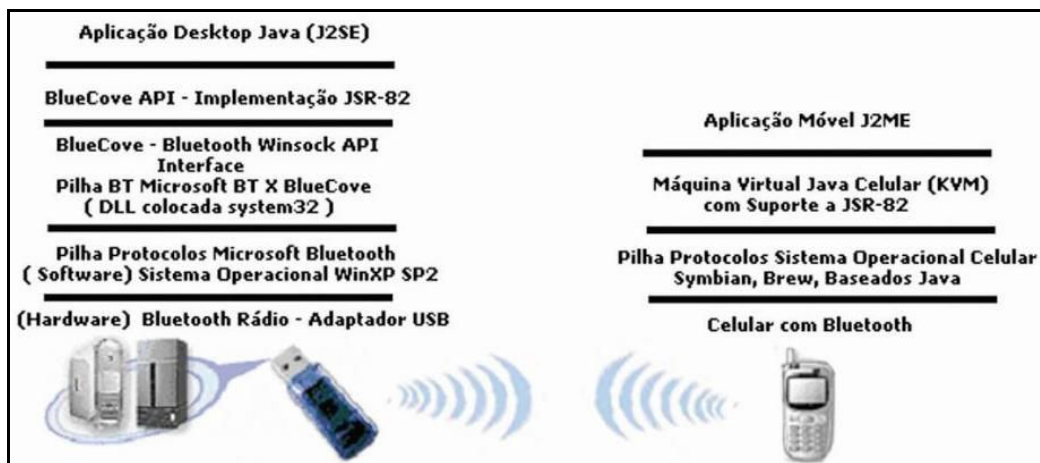
⁵ Fábrica: Padrão do projeto Marge na qual a classe fábrica é responsável por encapsular a criação de instâncias de outra classe

2.2.3 Bluecove

Bluecove é uma implementação da especificação JSR-82⁶ que pode ser utilizada na comunicação *Bluetooth* entre aplicações J2ME e J2SE. É utilizado a pilha de protocolo *Bluetooth* do Windows XP SP 2, ou seja, comunicação entre *Bluetooth* desktop e um dispositivo ambiente.

Para a comunicação *Bluecove* funcionar, seria necessário a utilização de uma *Dynamic-link library* (DLL) do Windows chamada *winsock*⁷ para fazer a interface entre a pilha *Bluetooth* da Microsoft e a implementação da API *Bluecove*. Porém, não havia implementação da pilha em Java, apenas C. Então foi criada a especificação JSR-82 a partir de uma API realizada via *Java Native Interface* (JNI⁸) para utilizar as bibliotecas em C.

A Figura 9 mostra um diagrama da arquitetura *Bluecove* na comunicação de uma MIDlet⁹ (MIDP 2.0) com suporte a especificação JSR-82 e uma aplicação J2SE usando a API *Bluecove* e a pilha *Bluetooth* do Windows XP SP2.



Fonte: Goelzer (2006).

Figura 9 - Diagrama da arquitetura Bluecove na comunicação com J2SE à J2ME.

⁶ JSR 82: *Java Specification Requests 82: Java™ APIs para Bluetooth*.

⁷ Winsock: especificação de *Sockets* para Windows. Define uma interface de programação de rede para Microsoft Windows que é baseada no paradigma "*socket*".

⁸ JNI: tecnologia que possibilita aplicações Java utilizarem bibliotecas nativas de código C.

⁹ MIDlet: um aplicativo em Java para dispositivos móveis, mais especificamente para a máquina virtual J2ME. Em geral são aplicações para serem executadas em celulares, como jogos entre outros.

2.3 MOBILE MARKETING

Mobile Marketing define-se pelo fato de pensar estratégias para envio de publicidade usando tecnologias móveis. Este meio de comunicação tem algumas características como comunicação a qualquer hora ou a interação e envolvimento (LANGE; SHROEDER, 2008).

Analisando o crescimento na venda de dispositivos móveis, percebe-se que esta mídia está cada vez mais presente na vida do consumidor. Segundo Prado (2006), com a aceitabilidade e a viabilidade do canal de comunicação móvel, profissionais em marketing começam a traçar táticas e estratégias para melhorarem seu retorno de investimento.

2.3.1 *Bluetooth* Marketing

Segundo Lins (2006), *Bluetooth* Marketing é um novo formato baseado em dispositivos que enviam conteúdo para celulares com a tecnologia *Bluetooth*. Os dispositivos com o intuito de interagir com celulares pesquisam, a uma curta distância, se há celulares com *Bluetooth* ativo. Ao identificar-se e com o pedido de autorização aceito (*opt-in*¹⁰), os dispositivos podem enviar ao consumidor vídeos, áudios, jogos, aplicativos, imagens, etc.

Segundo Mckenna (2002), uma empresa que torna constante sua presença ao consumidor, transmite uma experiência consistente e confiável a todo momento ao longo do tempo. Esta presença tem base em uma infraestrutura inteligente de informação, e permite ao profissional do marketing um diálogo constante com o consumidor. Hoje em dia as tecnologias de rede ajudam a criar essa presença constante com o consumidor.

Para gerenciar o pedido do consumidor, as empresas trabalham com as informações obtidas, de forma a registrar, monitorar e movimentar esses dados através da rede de suporte, onde há softwares que gerenciam essas informações. Os consumidores são atendidos de forma eficiente, segura e econômica com entrega dos serviços no ponto de presença.

Para Postma (1999), o relacionamento com o cliente é essencial para a prestação de um serviço. Uma comunicação personalizada possibilita atividades comerciais mais focadas e utilizadas como complemento para comunicação de massa, muitas vezes inclusive, as

¹⁰ *Opt-in*: termo empregado para as regras de envio de mensagens que definem que é proibido mandar e-mails comerciais/spam, a menos que exista uma concordância prévia por parte do destinatário.

substituindo-as por marketing um-a-um.

Marketing um-a-um, ou marketing individualizado é o estágio mais avançado do marketing de relacionamento (um conceito que tem por objetivo construir uma relação duradoura com o cliente). Marketing um-a-um tem o objetivo de individualizar e personalizar a empresa para cada cliente. Para isso é necessário, por exemplo, investimentos em tecnologias e desenvolvimento de produtos para resultar em políticas de relacionamento entre a empresa e o cliente (LIMEIRA, 2006).

Algumas determinações em relação ao tratamento e monitoramento dos clientes, segundo Kotler (1999), seriam, por exemplo, a pesquisa pela satisfação em relação aos produtos e serviços; encorajar *feedback*; de vez em quando fazer algo em especial que os agrade e nunca presumir que o cliente está garantido.

2.4 TRABALHO CORRELATOS

Existem projetos que desempenham papel semelhante ao desenvolvido neste trabalho. Dentre eles, foram escolhidos dois cujas características enquadram-se na mesma área de estudo. Foram selecionados os seguintes projetos: Aplicação Comercial para Celulares baseada em *M-Commerce* utilizando J2ME (SEDREZ, 2006) e Desenvolvimento de um Jogo *multiplayer* para celular (TRUPEL, 2008).

O projeto Aplicação Comercial para Celulares baseada em *M-Commerce* utilizando J2ME permite a compra de produtos pelo celular utilizando a tecnologia J2ME e o conceito M-Commerce (SEDREZ, 2006). Uma demonstração de uma compra pode ser visualizada na Figura 10.



Fonte: Sedrez (2006).

Figura 10 – Tela para informar itens e tela de compra efetuada

A aplicação permite o cadastro de produtos na aplicação servidora (módulo central) e

uma interação, através de MIDP, com o cliente e o vendedor do produto através de um celular.

O trabalho foi dividido em três módulos: o primeiro módulo foi uma aplicação que executa dentro do celular, onde se comunica com um Web Service em um servidor de aplicações; o segundo módulo foi executado na Web, onde são mantidas informações do consumidor, ponto de venda e produtos; o terceiro é o Web Service, o qual faz a comunicação entre o celular e o servidor.

Devido à complexidade do ponto de venda receber um sinal referente à solicitação de algum produto, este módulo do projeto foi implementado através do recebimento da informação através de e-mail. Já o módulo do comprador, realizado através de um emulador, foi realizado com sucesso.

O trabalho de Trupel (2008), Desenvolvimento de um jogo *multiplayer* para celular, usa a tecnologia Java juntamente com J2ME para a implementação de um jogo para celular usando a arquitetura de servidores. O objetivo é disponibilizar um cliente do jogo Batalha Naval, onde o servidor faz alguns processos como o de especificar porta para registro de novos jogadores ou associar jogadores aos adversários conforme a preferência escolhida por cada um. O jogo ocorre com no máximo quatro jogadores simultâneos e conta com a comunicação através de *sockets*. A Figura 11 ilustra uma tela com um cenário do jogo.



Fonte: Trupel (2008).

Figura 11 – Disposição dos personagens no cenário do jogo

Na conclusão o autor descreve que não teve êxito ao executar a aplicação em um celular, apenas simuladores. Ele relata dificuldades em relação a comunicação entre cliente e servidor, tendo a necessidade do desenvolvimento de um protocolo. Outra dificuldade foi o entendimento da plataforma J2ME.

Apesar das dificuldades, os objetivos deste trabalho foram alcançados. Dois jogadores conseguiram jogar de forma individual através de simuladores. Outros quatro jogadores conseguiram jogar através de simuladores, sendo que para esta opção de jogo é tratada no servidor como um jogo em duplas.

3 DESENVOLVIMENTO

As seções seguintes descrevem a especificação, implementação e a operacionalidade da aplicação mobile com integração entre J2ME e J2SE através do *Bluetooth*. A operacionalidade é apresentada de acordo com o estudo de caso de uma aplicação voltada à uma lanchonete. Por fim, são indicados os resultados obtidos com este trabalho.

3.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

Uma aplicação mobile marketing utilizando comunicação *Bluetooth* necessita de um servidor disponibilizando dados requisitados conforme cada serviço escolhido pelo usuário da aplicação cliente. Antes ainda, o cliente precisa estar cadastrado no servidor para que a aplicação servidora envie dados à aplicação cliente. A troca de mensagem pode ser efetuada através de um *layout* conhecido tanto pela aplicação servidora quanto a aplicação cliente.

Na sequência são apresentados os Requisitos Funcionais (RF) e Requisitos Não Funcionais (RNF) que são atendidos pelo módulo servidor e o módulo cliente. No Quadro 2 podem ser observados os RNF e no Quadro 3 os RF.

Requisitos Não Funcionais (RNF)
RNF01 - usar comunicação por meio <i>Bluetooth</i>
RNF02 - a comunicação <i>Bluetooth</i> deve ser feita através da biblioteca <i>Bluecove</i>
RNF03 - utilizar framework para desenvolvimento Marge
RNF04 - usar linguagem de programação Java

Quadro 2 - Requisitos não funcionais

Requisitos Funcionais (RF)	Caso de Uso (UC)
RF01 - O módulo servidor deve pesquisar os dispositivos que estão com <i>Bluetooth</i> ativo	UC01
RF02 - O módulo servidor deve permitir o cadastro de usuários	UC02
RF03 - O módulo servidor deve permitir o cadastro de cardápio	UC03
RF04 - O módulo servidor deve permitir o cadastro de promoções	UC04
RF05 - O módulo servidor deve permitir o cadastro de campanhas	UC05
RF06 - O módulo servidor deve permitir o cadastro de enquetes	UC06
RF07 - O módulo servidor deve permitir o controle de pedidos	UC07
RF08 - O módulo servidor deve apresentar a quantidade de pedidos por usuário	UC08
RF09 - O módulo servidor deve apresentar a quantidade de campanhas recebidas por usuário	UC08
RF10 - O módulo servidor deve apresentar a quantidade de enquetes respondidas por usuário	UC08
RF11 - O módulo cliente deve permitir ao usuário escolher o serviço que será prestado (Pedido, Promoção ou Enquete)	UC12
RF12 - O módulo servidor, quando requisitado o serviço pedido, deve enviar os dados de pedidos ao usuário	UC09
RF13 - O módulo cliente deve enviar ao servidor os dados do pedido efetuado	UC13
RF14 - O módulo servidor, quando requisitado o serviço promoção, deve enviar os dados das promoções cadastradas para o usuário solicitante	UC10
RF15 - O módulo cliente deve enviar ao servidor a confirmação de leitura da promoção	UC14
RF16 - O módulo servidor, quando requisitado o serviço enquete, deve enviar os dados das enquetes cadastradas para o usuário solicitante	UC11
RF17 - O módulo cliente deve enviar ao servidor as respostas da enquete	UC15

Quadro 3 - Requisitos funcionais

3.2 ESPECIFICAÇÃO

Na sequência é apresentada a especificação da aplicação, que foi modelada na ferramenta *Enterprise Architect*. Foram utilizados conceitos da orientação a objetos e a *Unified Modeling Language* (UML) para a criação dos diagramas de casos de uso, classe e de sequência.

3.2.1 Diagramas de casos de uso

Na aplicação existem dois cenários. O primeiro é o servidor e o segundo o cliente. A seguir serão apresentados os diagramas de caso de uso do servidor e do cliente, detalhando os principais.

3.2.1.1 Módulo servidor

A Figura 12 ilustra os casos de uso do módulo servidor executados pelo usuário. Estes casos de uso são referentes a cadastros e visualização dos dados.

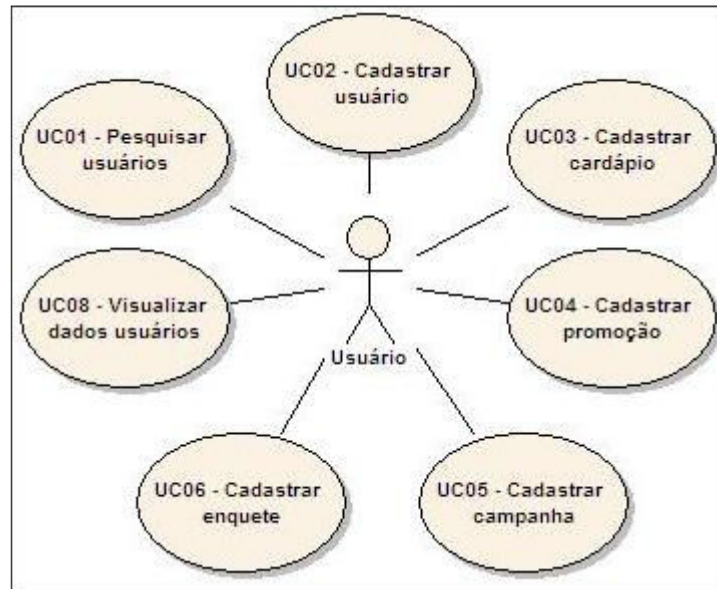


Figura 12 - Diagrama de casos de uso do módulo servidor executados pelo usuário

A Figura 13 mostra os casos de uso também do servidor, porém são executados pela aplicação quando recebem alguma requisição ou resposta do cliente. Os casos de uso detalhados são: Enviar dados do pedido, Enviar dados das promoções e Enviar dados das enquetes.

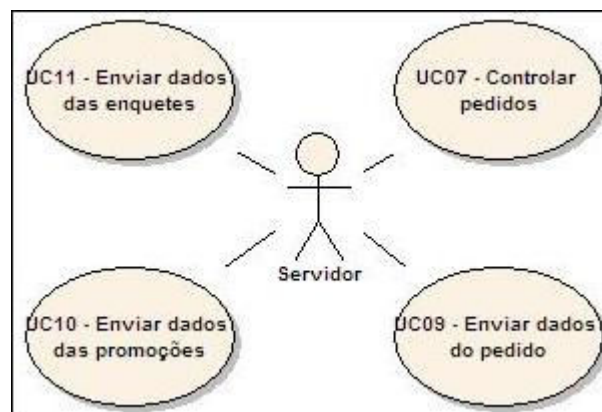


Figura 13 - Diagrama de casos de uso do módulo servidor executados pela aplicação

O primeiro caso de uso, Enviar dados do pedido (Quadro 4), representa as ações do módulo servidor ao receber uma solicitação do módulo cliente para envio de pedido. Ele possui, além do cenário principal, um cenário alternativo e dois cenários de exceção informando possíveis erros encontrados durante o envio.

UC09 – Enviar dados do pedido: permite o envio dos dados pertinentes a um pedido	
Pré-condições	Ter pontos de atendimento e cardápio cadastrado e também receber uma requisição de pedido vindo do módulo cliente.
Cenário principal	01) O módulo servidor monta uma string com os pontos de atendimento. 02) O módulo servidor monta os itens a serem enviados conforme os pedidos do cliente. 03) O módulo servidor inclui o MAC do cliente, a requisição e os separadores (#). 04) O módulo servidor transforma a mensagem em <i>byte</i> . 05) O módulo servidor procura um canal de comunicação. 06) O módulo servidor envia a mensagem.
Fluxo alternativo 01	No passo 02, o módulo servidor monta os itens a serem enviados conforme os pedidos já recebidos: 02.01) quando o módulo servidor recebe um pedido, ele abre a tabela de pedidos recebidos e incrementa o item em relação ao cliente que o solicitou. 02.02) se for o primeiro pedido do cliente, os itens são enviados conforme ordem do cadastro.
Exceção 01	No passo 03, caso os dados solicitados não tenham sido cadastrados, é apresentada ao usuário uma mensagem solicitando que os dados sejam cadastrados.
Exceção 02	No passo 04, caso os dados solicitados não tenham sido cadastrados, é apresentada ao usuário uma mensagem solicitando que os dados sejam cadastrados.
Pós-condições	Dados do pedido enviado.

Quadro 4 - Detalhamento do caso de uso *Enviar dados do pedido*

O segundo caso de uso, *Enviar dados das promoções* (Quadro 5), representa as ações do módulo servidor ao receber uma solicitação do módulo cliente para envio de promoções. Ele possui, além do cenário principal, um cenário alternativo e um cenário de exceção informando possíveis erros encontrados durante o envio.

UC10 – Enviar dados das promoções: permite o envio dos dados pertinentes a promoções	
Pré-condições	Ter campanha cadastrada e também receber uma requisição de promoção vindo do módulo cliente.
Cenário principal	01) O módulo servidor monta uma string com promoção referente ao cliente que a solicitou. 02) O módulo servidor inclui o MAC do cliente, a requisição, o nome do cliente e os separadores (#). 03) O módulo servidor transforma a mensagem em <i>byte</i> . 04) O módulo servidor procura um canal de comunicação. 05) O módulo servidor envia a mensagem.
Fluxo alternativo 01	No passo 01, se a promoção encontrada para o cliente que a requisitou já foi enviada e a promoção não deve se repetir, o servidor não envia novamente.
Exceção 01	No passo 01, caso não exista promoções para o cliente que a requisitou, o servidor envia uma mensagem ao cliente avisando que não há promoções.
Pós-condições	Dados das promoções enviados.

Quadro 5 - Detalhamento do caso de uso *Enviar dados das promoções*

O terceiro caso de uso, *Enviar dados das enquetes* (Quadro 6), representa as ações do módulo servidor ao receber uma solicitação do cliente para envio de enquetes. Ele possui, além do cenário principal, dois cenários alternativos e um cenário de exceção informando

possíveis erros encontrados durante o envio.

UC11 – Enviar dados das enquetes: permite o envio dos dados pertinentes a enquetes	
Pré-condições	Ter enquete cadastrada e também receber uma requisição de enquete vindo do cliente.
Cenário principal	01) O módulo servidor monta uma string com enquetes referentes ao cliente que a solicitou 02) O módulo servidor inclui o MAC do cliente, a requisição, o nome do cliente e os separadores (#) 03) O módulo servidor transforma a mensagem em <i>byte</i> . 04) O módulo servidor procura um canal de comunicação. 05) O módulo servidor envia a mensagem.
Fluxo alternativo 01	No passo 01, se a enquete encontrada para o cliente que a requisitou já foi enviada, o servidor não envia novamente.
Fluxo alternativo 02	No passo 01, a enquete encontrada para o cliente pode ser de dois tipos: 01.01: Seleção única: o cliente poderá escolher apenas um item; 01.02: Multi-seleção : o cliente poderá escolher um ou mais itens.
Exceção 01	No passo 01, caso não exista enquetes para o cliente que a requisitou, o servidor envia uma mensagem ao cliente avisando que não há pesquisas.
Pós-condições	Dados das enquetes enviados.

Quadro 6 - Detalhamento do caso de uso Enviar dados das enquetes

3.2.1.2 Módulo cliente

A Figura 14 ilustra os casos de uso do módulo cliente. Porém, o caso de uso mais pertinente é o Enviar dados do pedido efetuado.

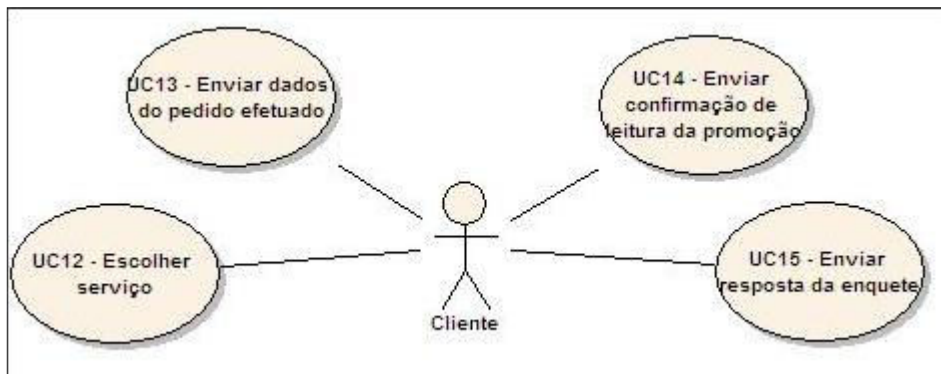


Figura 14 - Diagrama de casos de uso da aplicação módulo cliente

O caso de uso, Enviar dados do pedido efetuado (Quadro 7), representa as ações do módulo cliente no processo de informar os dados do pedido para envio ao servidor. Ele possui, além do cenário principal, um cenário alternativo e dois cenários de exceção informando possíveis erros encontrados durante o envio.

UC13 – Enviar dados do pedido efetuado: permite o envio dos dados pertinentes a um pedido efetuado	
Pré-condições	Ter pontos de atendimento e cardápio recebido pela aplicação servidor.
Cenário principal	01) O módulo cliente verifica se a mensagem recebida é realmente destinada a ele. 02) O módulo cliente verifica se o tipo de requisição é um pedido. 03) O módulo cliente monta uma tela com os pontos de atendimentos recebidos. 04) O usuário escolhe o ponto de atendimento e clica em “Continuar”. 05) O módulo cliente monta uma tela com o cardápio recebido. 06) O usuário escolhe a quantidade de cada item que desejar e clica em “Continuar”. 07) O módulo cliente monta uma tela com o pedido do usuário, valor total e a possibilidade do usuário solicitar ajuda. 08) O usuário clica em “Finalizar”. 09) O módulo cliente transforma a mensagem em <i>byte</i> . 10) O módulo cliente procura um canal de comunicação. 11) O módulo cliente envia a mensagem 12) O módulo cliente espera até 15 segundos uma mensagem de confirmação de recebimento do módulo servidor. 13) O módulo cliente apresenta ao usuário a mensagem enviada.
Fluxo alternativo 01	No passo 07, caso o usuário deseja alterar o pedido: 07.01: clica em “Voltar” 07.02: retorna para o passo 06.
Exceção 01	No passo 06, caso o usuário não escolha nenhum item, não poderá seguir com o pedido.
Exceção 02	No passo 12, se o tempo limite de 15 segundos for ultrapassado, o módulo cliente emite uma mensagem de erro no envio da mensagem ao módulo servidor.
Pós-condições	Dados do pedido enviado com sucesso.

Quadro 7- Detalhamento do caso de uso Enviar dados do pedido efetuado

3.2.2 Diagrama de classes

O diagrama de classes fornece uma visão de como as classes estão estruturadas e relacionadas. De forma a facilitar a estruturação e a relação entre elas, são apresentados dois diagramas de classes no módulo servidor e dois diagramas de classes no módulo cliente.

3.2.2.1 Módulo servidor

O módulo servidor apresenta várias classes divididas em cinco pacotes. Três destes pacotes implementam a manipulação dos dados e a interface com o usuário, e são definidos no Quadro 8.

Definições dos pacotes de manipulação de dados e de interface com usuário	
PACOTE	DEFINIÇÃO
EstruturaTabelas	Armazena classes contendo a estrutura de cada tabela da aplicação, como Usuário, Pedido, Promoção entre outras. As tabelas são armazenadas em disco através do formato <i>Extensible Markup Language (XML)</i> .
FluxoDados	Contem as classes que controlam a persistência das informações em disco. São implementados métodos que gravam, alteram e excluem os registros armazenados em disco.
InterfaceGrafica	Este pacote possui classes que definem as telas e o controle de cada componente em cada tela. Aqui é definido a ação de cada botão, tabela e outros componentes de interação diretamente com o usuário.

Quadro 8 - Definições dos pacotes de manipulação de dados e de interface com usuário

A Figura 15 ilustra as classes do pacote de comunicação *Bluetooth*. A classe `PesquisaDispositivos`, que está abaixo das classes do pacote `InterfaceGrafica`, cria uma instância da classe `Inquiry`, onde há também uma instância da classe `InquiryListener`, *interface* do *framework* `Marge` que contém métodos relativos ao processo de busca. Estas classes são responsáveis pela busca de dispositivos com *Bluetooth* ativo. Através deste processo, é possível visualizar o endereço de *Media Access Control (MAC)*, que é um endereço único, como o que acontece com as placas de rede. Também é possível identificar o apelido dispositivo.

A classe `RFCOMMServer` tem a função de estabelecer conexão com dispositivos cliente, enviar e receber mensagens e também tratar os possíveis erros. Esta classe é implementada através das interfaces `ConnectionListener` e `CommunicationListener`, também do *framework* `Marge`.

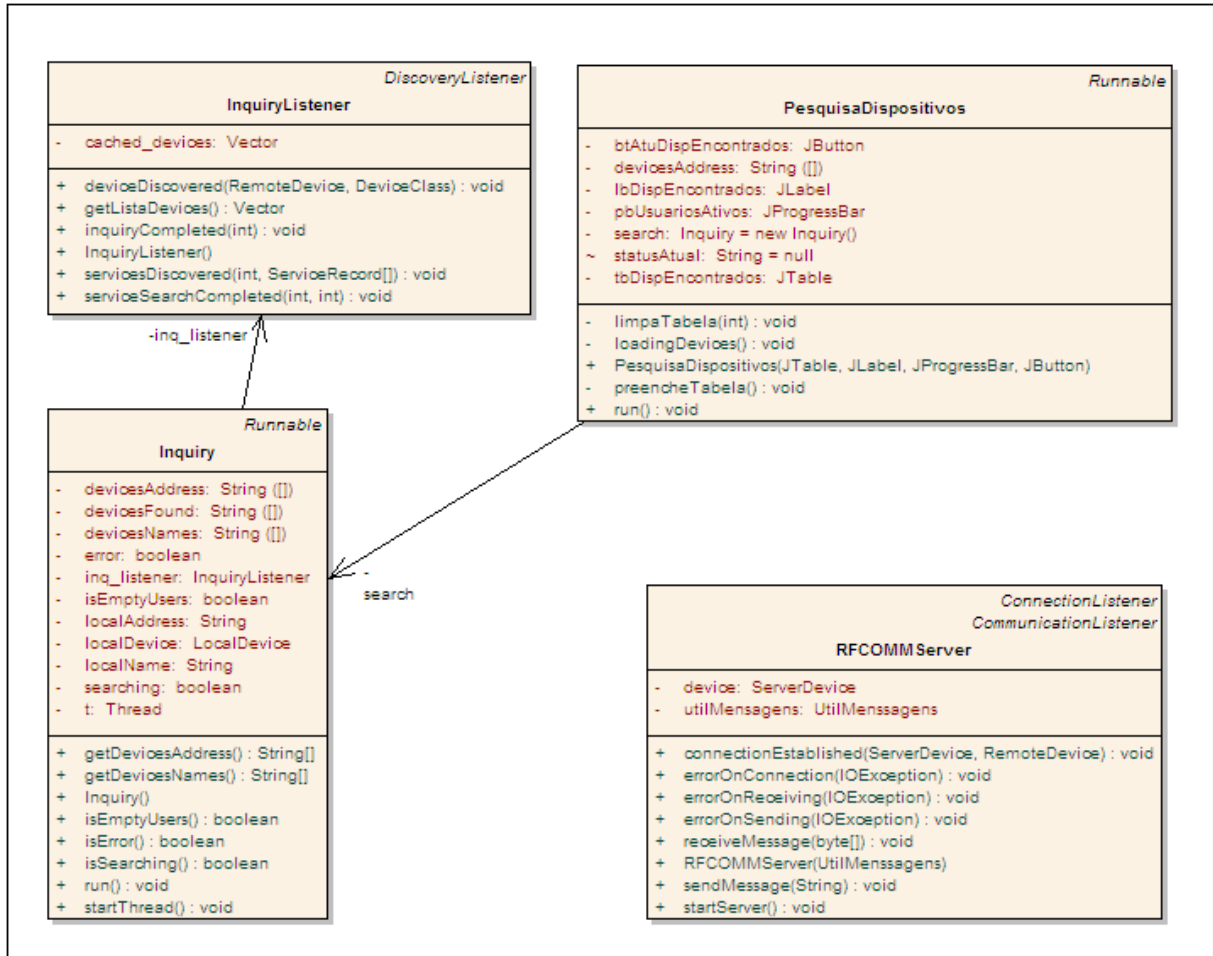


Figura 15 - Pacote com estrutura da comunicação *Bluetooth*

Para o tratamento das mensagens recebidas pelo servidor, há um pacote que faz a manipulação de cada mensagem, fazendo o tratamento necessário, dependendo da requisição. A Figura 16 detalha um pouco mais as classes do pacote.

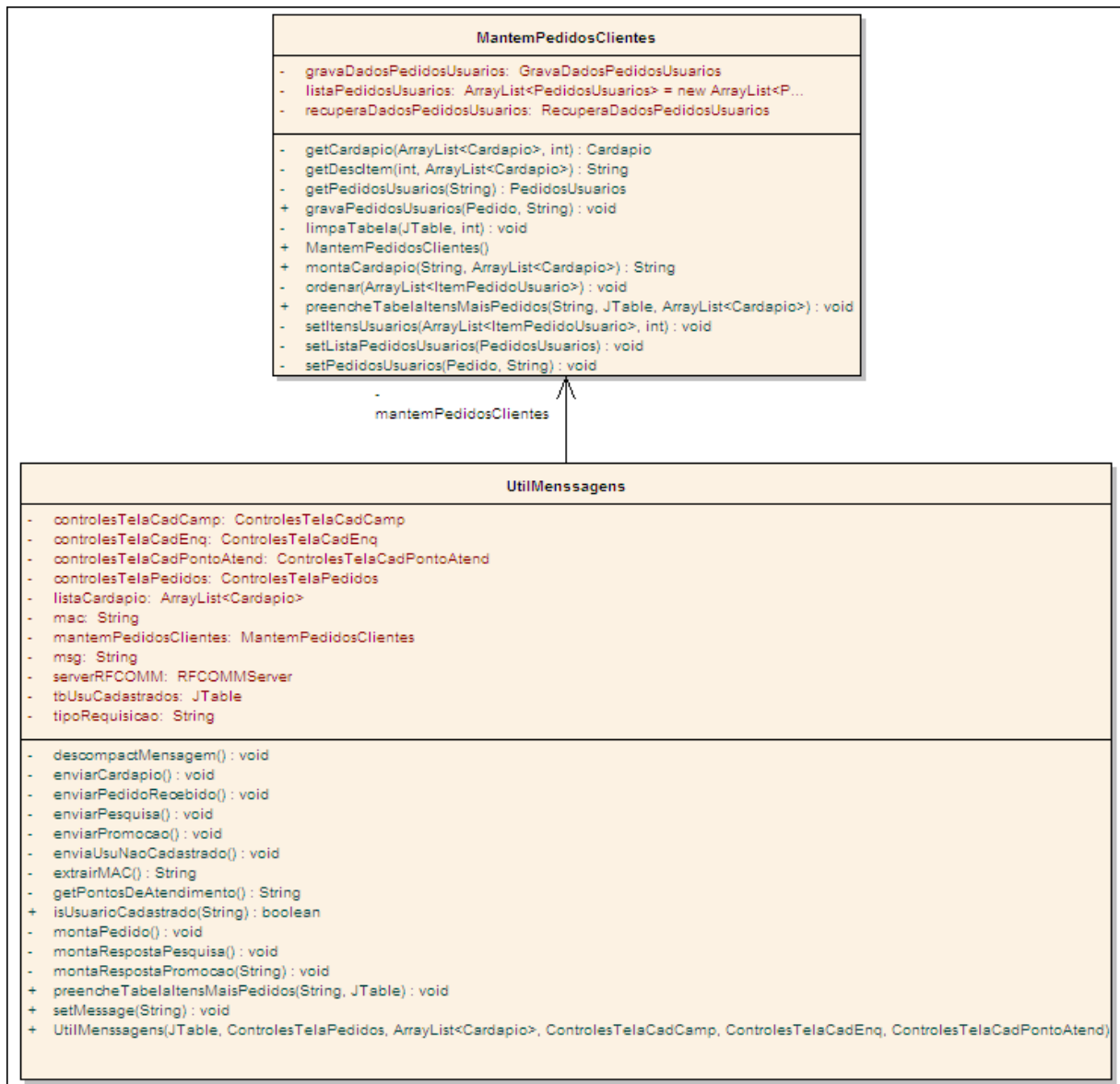


Figura 16 - Classes do pacote que trata as mensagens recebidas via *Bluetooth*

A classe `UtilMenssagens` controla todo o fluxo de dados da comunicação. O método `descompactaMensagem()` identifica o usuário que está enviando e a requisição solicitada. Como o processo de Pedido tem algumas funcionalidades a mais, como controle de pedidos por usuário e os itens mais pedidos, foram separados alguns tratamentos para que o entendimento do processo se tornasse menos complexo. Para isso foi criada a classe `MantemPedidosClientes`, onde há o controle dos pedidos de cada cliente e a quantidade de cada item solicitado.

3.2.2.2 Módulo cliente

O módulo cliente apresenta várias classes divididas em dois pacotes. No primeiro pacote, `bluetoothmenu` (Figura 17), se encontram as classes das telas e as classes da comunicação. As classes `InquiryMenu` e `Apresentacao` estabelecem a conexão, enquanto a classe `WaitMessage` tem a função de monitorar cada mensagem que chega do servidor. A partir do momento que é identificado o tipo de requisição, é chamada a classe correspondente ao serviço. As demais classes do pacote criam as telas com interação com o usuário.

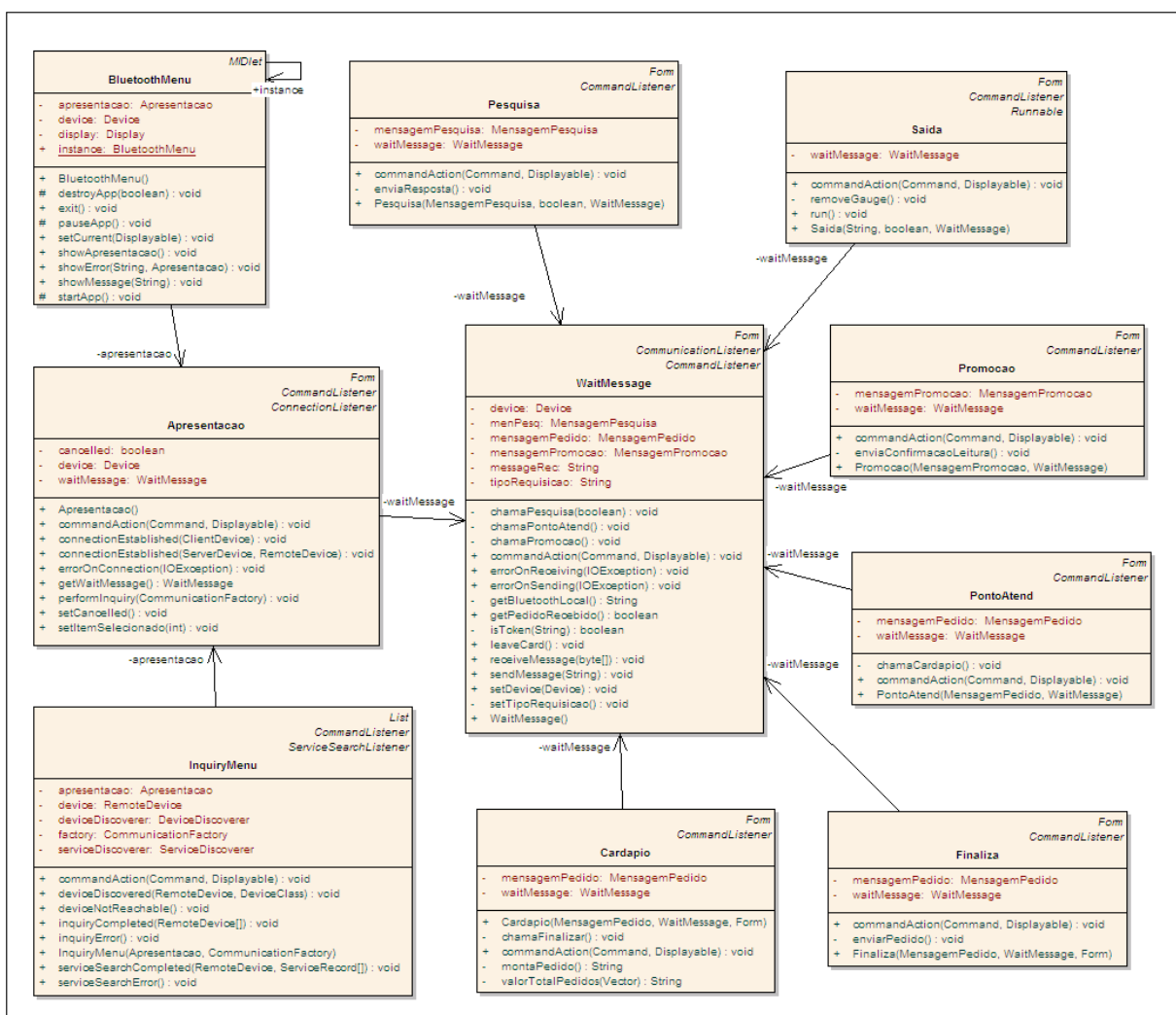


Figura 17 - Pacote contendo a estrutura das telas e da comunicação

O pacote `TratamentoDados` (Figura 18), manipula as mensagens recebidas do servidor e as que serão enviadas. O tratamento das mensagens é feito conforme a solicitação de serviço que o usuário definiu. Cada classe controla dados de um serviço específico. Os pedidos são armazenados nas classes `MensagemPedido`, `ItemDisp` e `PontoAtendDisp`. Cada uma dessas

classes são utilizadas no momento em que as telas do pedido são apresentadas aos usuários. Da mesma forma em que a classe `MensagemPesquisa` descompacta e armazena as enquetes e a classe `MensagemPromocao` descompacta e armazena as promoções.

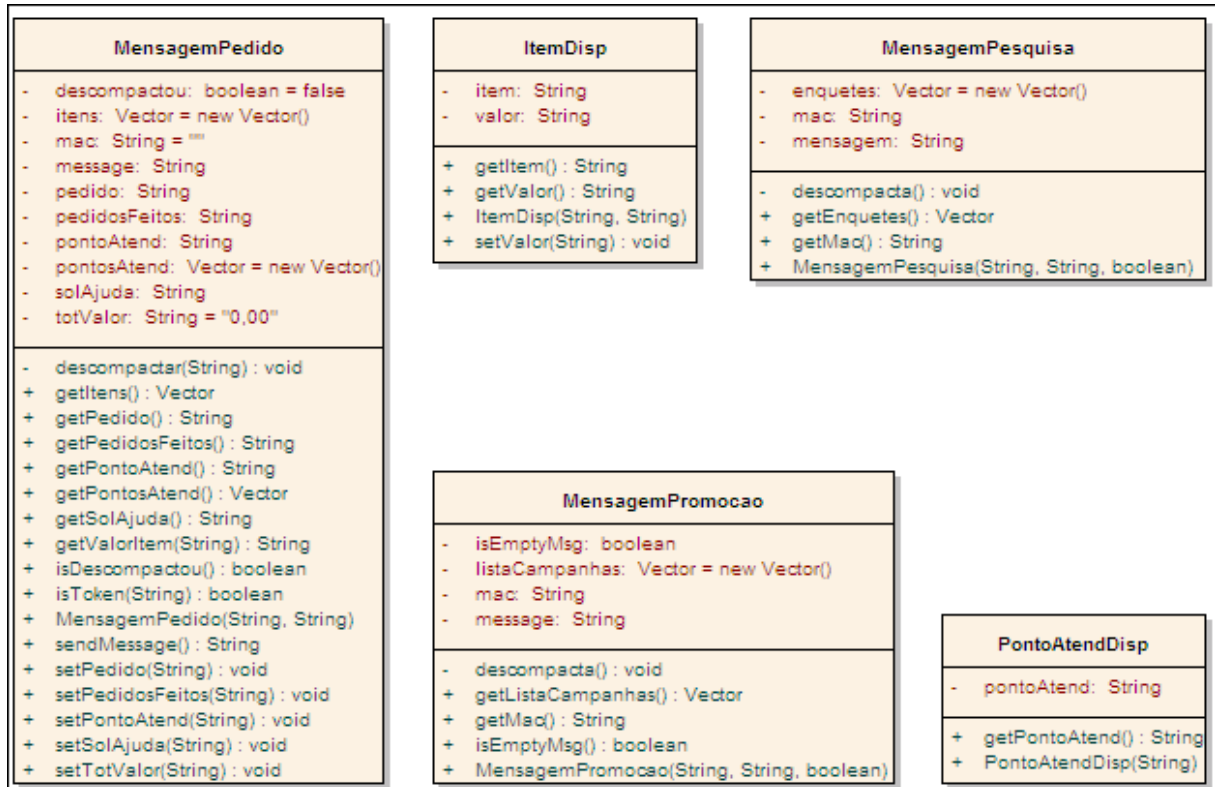


Figura 18 - Pacote contendo as classes manipulam as mensagens

3.2.3 Diagrama de sequência

Esta seção apresenta os diagramas de sequência que representam o conjunto de passos que o programa executa para realizar determinada tarefa, com base nas ações do usuário.

A Figura 19 apresenta a sequência de chamada de métodos realizada pelo sistema para que o usuário faça um pedido. Este diagrama corresponde ao caso de uso `Enviar dados do pedido efetuado` do módulo cliente. A aplicação estabelece uma conexão e o usuário solicita o cardápio e espera a resposta do servidor. Quando a resposta chega, a aplicação descompacta a mensagem e chama a tela de pontos de atendimento. O usuário seleciona o ponto de atendimento e a aplicação mostra a tela de itens. Depois de escolher os itens a aplicação informa os itens escolhidos e o valor final. O usuário fecha o pedido e a aplicação envia o pedido ao servidor.

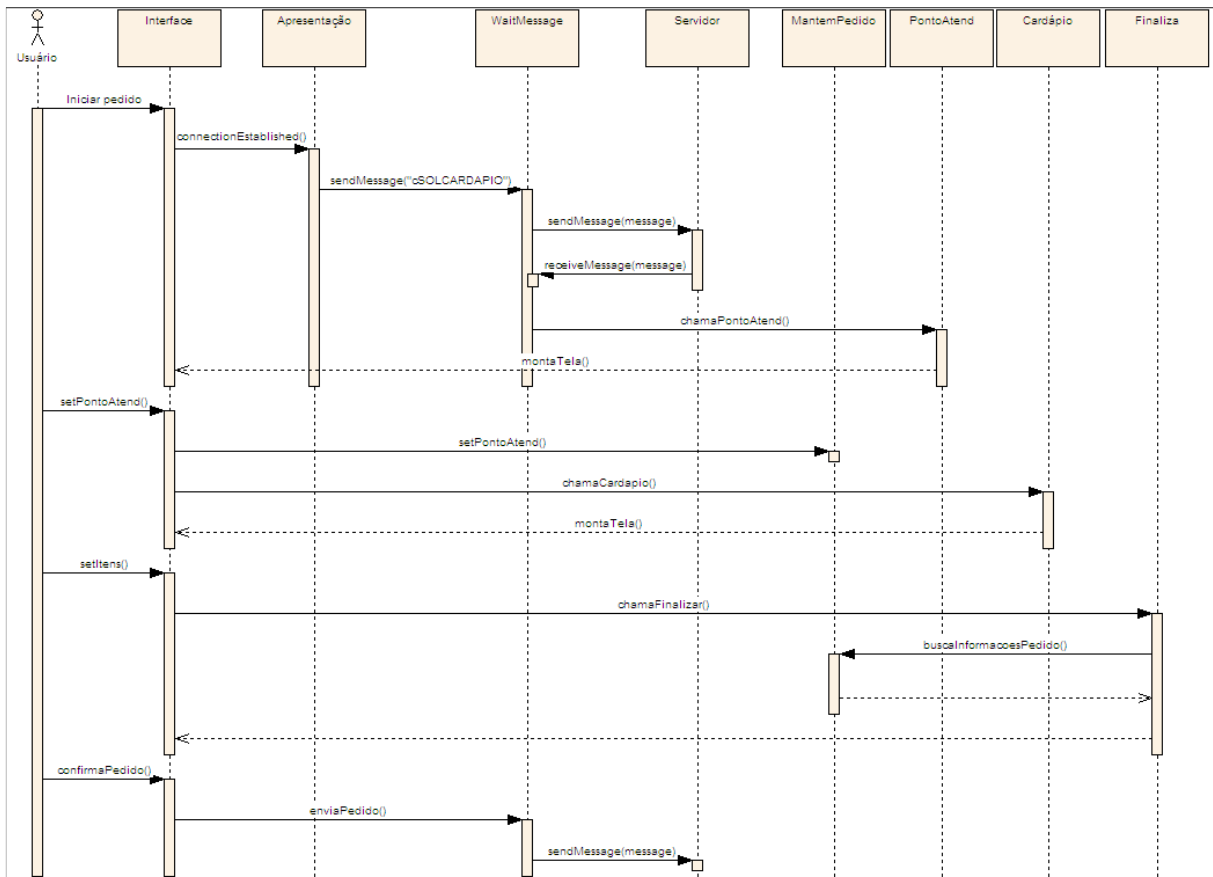


Figura 19 - Diagrama de sequência para enviar pedido

A Figura 20 apresenta a sequência de chamada de métodos realizada pelo sistema no módulo cliente para que o usuário receba promoções. A aplicação estabelece uma conexão e o usuário solicita promoções e aguarda a resposta do servidor. Quando a aplicação recebe uma resposta do servidor, as telas são montadas e o usuário pode visualizar as promoções recebidas.

A Figura 21 apresenta a sequência de chamada de métodos realizada pelo sistema no módulo cliente para que o usuário receba enquetes. A aplicação estabelece uma conexão e o usuário solicita enquetes e aguarda a resposta do servidor. Quando a aplicação recebe uma resposta do servidor, as telas são montadas e o usuário pode visualizar as enquetes recebidas.

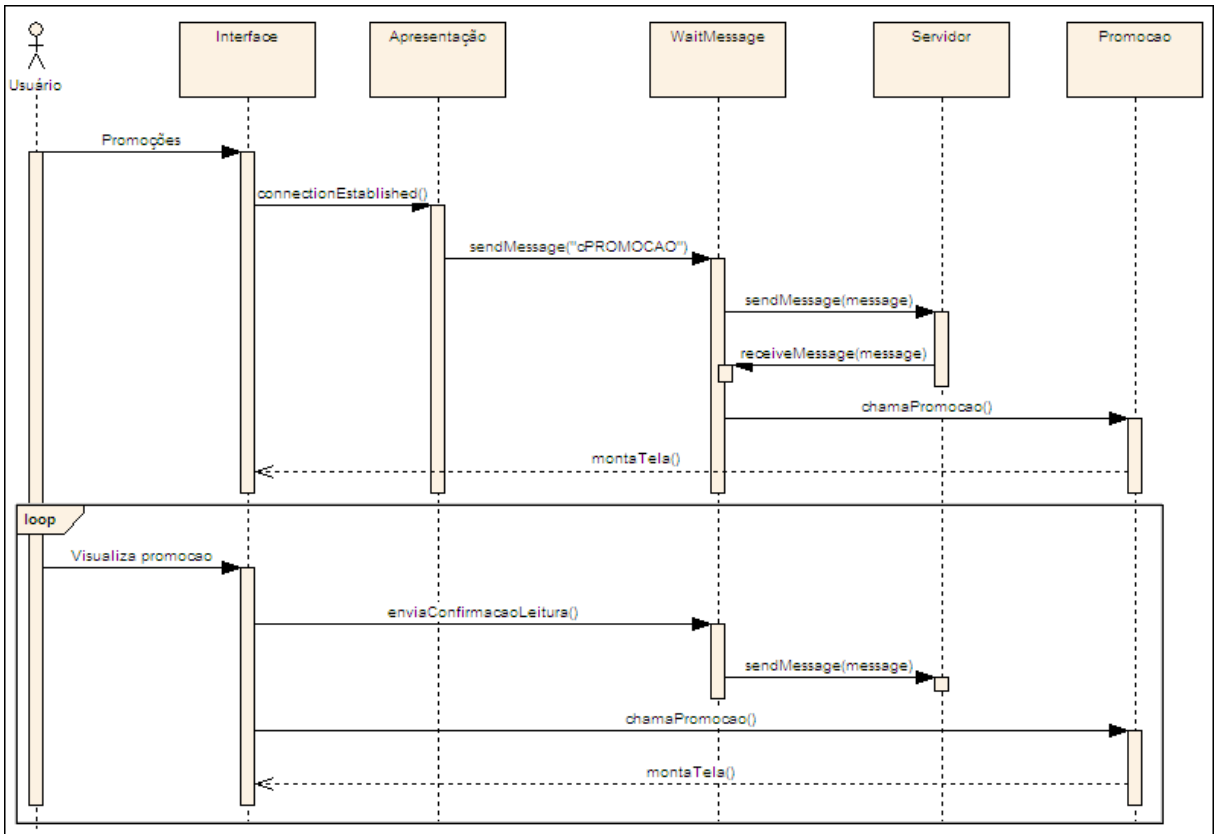


Figura 20 - Diagrama de sequência para cliente receber promoções

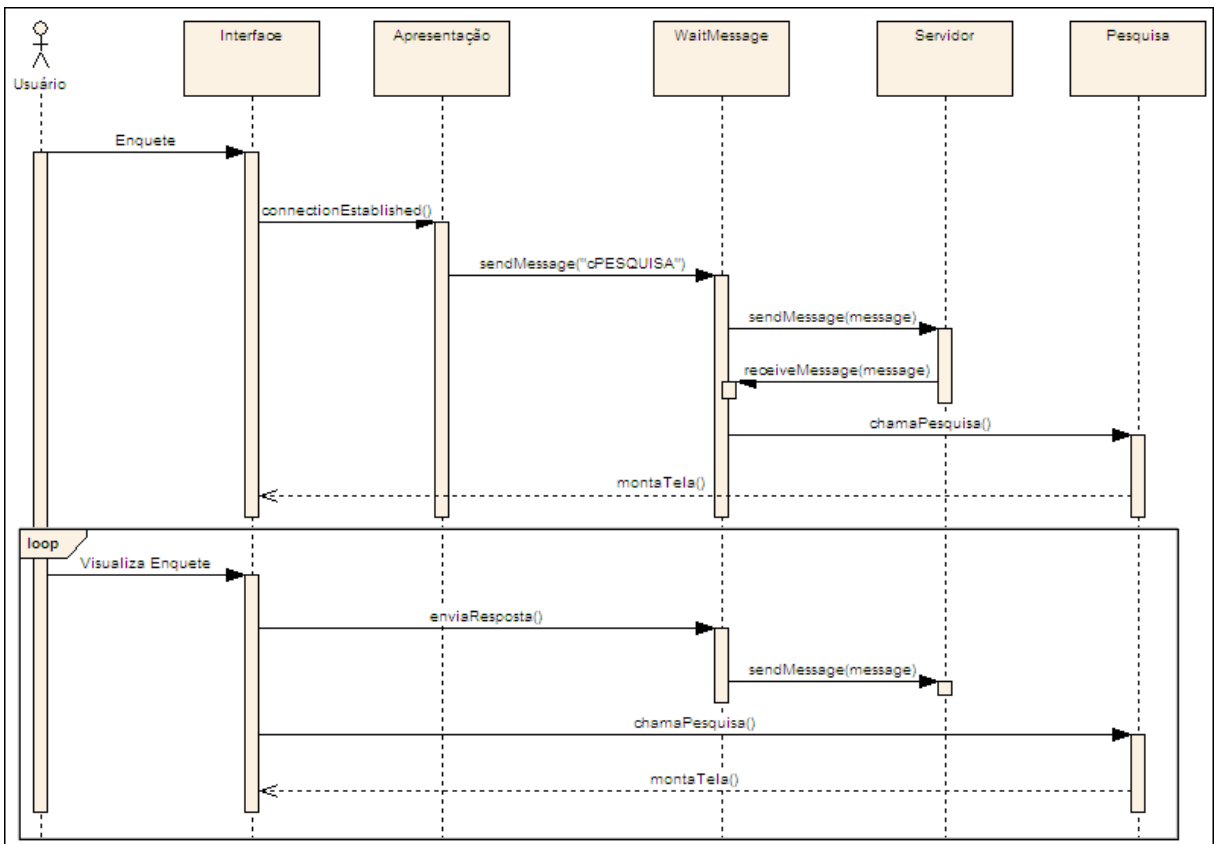


Figura 21 - Diagrama de sequência para cliente receber enquetes

3.3 IMPLEMENTAÇÃO

Nesta seção são apresentadas informações sobre as técnicas e ferramentas utilizadas para a implementação da aplicação, bem como o processo de implementação.

3.3.1 Técnicas e ferramentas utilizadas

A aplicação foi implementada na linguagem Java, seguindo o paradigma da orientação a objetos. Para a codificação foram utilizados dois ambientes de desenvolvimento (Eclipse e NetBeans), o *framework* Marge e uma biblioteca chamada *Blucove* (os dois últimos já discutidos no capítulo 2.2.2 e 2.2.3).

O Eclipse (versão 3.5.0) foi utilizado para a implementação de toda a camada de modelo e de controle. Já o NetBeans (versão 6.5.1) foi utilizado na implementação da camada de visão, por permitir uma ágil prototipação das janelas.

3.3.2 Desenvolvimento da aplicação

A aplicação foi desenvolvida através de dois módulos: o módulo servidor e o módulo cliente. O módulo servidor foi desenvolvido para ser executado em um desktop e o módulo cliente será executado através da aplicação instalada dentro de um celular comum.

3.3.2.1 Módulo servidor

Como a maioria dos usuários não conhece o endereço MAC do seu dispositivo, a aplicação servidor tem a opção de descobrir os usuários e seus respectivos endereços MAC através do processo de pesquisa por usuários, criando a classe `PesquisaDispositivos`. Através desta, a aplicação faz uma busca pelos dispositivos com *Bluetooth* ativo e ao alcance do servidor.

Com os dados recuperados através da pesquisa, é possível efetuar o cadastro do usuário. Lembrando apenas que, o usuário conhecendo o endereço MAC de seu dispositivo,

não há a necessidade de uma pesquisa. O cadastro se faz através da classe `Usuário`, onde é definida a estrutura do arquivo “usuários.xml”. Em seguida, as classes `GravaDadosUsuario` e `RecuperaDadosUsuario` implementam a persistência dos dados em disco. Através da classe `ControlesTelaCadUsu` a classe de interface gráfica faz a manipulação dos dados.

Os cadastros de “Cardápio”, “Promoção” e “Ponto de atendimento” são efetuados da mesma forma. Há uma classe contendo a estrutura do arquivo, a persistência dos dados, e por fim, uma classe de controle, onde a interface gráfica faz a manipulação dos dados.

As classes que controlam os cadastros de “Enquetes” e “Campanhas” são muito parecidas com as já analisadas. Porém, o diferencial é que, cada uma delas tem a opção de vincular clientes.

O cadastro de pedidos é efetuado através de requisições vinda de um cliente. A classe `RFCOMMServer` fica monitorando através do método `receiveMessage(message)` as mensagens recebidas. Quando uma mensagem chega ao servidor, ela é enviada para a classe `UtilMensagens`, onde há o tratamento da mensagem. A primeira verificação é se o cliente já é cadastrado, pra em seguida ser chamado o método `descompactMensagem()`, onde a mensagem é tratada. O método `descompactMensagem()` será detalhado mais adiante.

Se a mensagem que chega é um início de um pedido, o servidor monta o cardápio conforme os pedidos que o cliente já efetuou e envia para o cliente. No Quadro 9 há um exemplo de solicitação de cardápio, ação que inicia um pedido.

```
#ENDERECO MAC DO CLIENTE#TIPO DE REQUISIÇÃO#
#0021FE738301#cSOLCARDAPIO
```

Quadro 9 - *String* compactada recebida de um cliente

No Quadro 10 ilustra um pedido enviado do servidor para o cliente.

```
#ENDERECO MAC DO CLIENTE#TIPO DE REQUISIÇÃO#PONTOS ATENDIMENTO#DESC
PROD 1$VALOR|DESC PROD 2$VALOR

#0021FE738301#cPEDIDO#Mesa 1&Mesa 2&Mesa 3&Mesa 4#448 - Coca Cola$3,00|445 - Cerveja
Skol$5,00|449 - Batata Frita$5,00
```

Quadro 10 - *String* enviada do servidor para um cliente

Quando o servidor recebe um pedido, este é descompactado e o processo de gravar é executado. O Quadro 11 ilustra um pedido recebido de um cliente.

```
#ENDERECO MAC DO CLIENTE#TIPO DE REQUISIÇÃO#PONTO
ATENDIMENTO#DESC PROD 1$QTD$VALOR# DESC PROD 1$QTD$VALOR&SOL
AJUDA

#0021FE738301#cPEDIDO#Mesa 2#448 - Coca Cola$02$3,00#449 - Batata
Frita$01$5,00&Nao
```

Quadro 11 - *String* enviada por um cliente

O Quadro 12 demonstra mensagens da forma com que são enviadas promoções e enquetes.

PROMOÇÃO

Recebido do cliente

#0021FE738301#cPROMOCAO

Enviado ao cliente - Neste momento o cliene é tratado pelo nome

#0021FE738301#cPROMOCAO#Rafael#Inauguração#Promoção de inauguração

PESQUISA

Recebido do cliente

#0021FE738301#cPESQUISA

Enviado ao cliente - Neste momento o cliene é tratado pelo nome

//Indica que é multiselecao

#0021FE738301#cPESQUISA#Rafael#1#MULTI#Forma de pagamento de sua preferência#Dinheiro#Cartão#Cheque

Recebido do cliente

#0021FE738301#cRPESQUISA#1#Dinheiro#Cartão

Quadro 12 - Troca de mensagens no serviço de promoção e pesquisa

Para inicializar uma conexão, é necessário ativar o servidor RFCOMM. No Quadro 13 demonstra a implementação da inicialização do servidor.

```
//inicia servidor
public void startServer() {
    //CommunicationFactory: Interface que define métodos para criação de
    //dispositivos clientes e servidores.
    //RFCOMMCommunicationFactory: Classe que define métodos para criação
    //de dispositivos clientes e servidores que usam o protocolo RFCOMM
    //para comunicação.
    //Protocolo RFCOMM emula uma porta serial
    CommunicationFactory factory = new RFCOMMCommunicationFactory();

    //ServerConfiguration: Classe que representa a configuração de
    //um servidor
    //Um ServerConfiguration deve obrigatoriamente receber, no construtor,
    //um CommunicationListener, que é uma interface pertencente ao pacote
    //net.java.dev.marge.communication que define métodos para tratamento
    //das seguintes situações: quando uma mensagem é recebida; quando
    //ocorre um erro na recepção de uma mensagem; quando ocorre um erro
    //no envio de uma mensagem
    ServerConfiguration config = new ServerConfiguration(this);

    //Limita o número máximo de conexões
    config.setMaxNumberOfConnections(7);

    //waitClients, recebe uma instância que implementa a interface
    //ConnectionListener, que será notificada quando alguma conexão for
    //estabelecida, retornando, então, os dispositivos envolvidos,
    //tanto o servidor, como o cliente.
    factory.waitClients(config, this);
}
```

Quadro 13 - Inicialização do servidor na classe RFCOMMServer

Para o processo de envio de mensagens, a aplicação procura os canais de comunicação e envia a mensagem a cada um deles. Os canais de comunicação, conforme estipulado por Miller (2001), cada rede *piconet* contem até sete dispositivos mais o servidor. Com isso, cada

mensagem enviada acaba se tornando uma mensagem de *broadcast*¹¹ sendo que, cada aplicação cliente verifica se a mensagem é destinada a ela. No Quadro 14 é ilustrada a implementação do envio de mensagem pela classe `Device`, dentro do *framework* Marge. O método `sendMessage()` dentro da classe `RFCOMMServer` faz a chamada do método `send()` do *framework*.

```
public void send(byte[] message) {
    for (int i = 0; i < channels.size(); i++) {
        try {
            this.getChannel(i).send(message);
        } catch (IOException e) {
            this.communicationListener.errorOnSending(e);
            this.channels.removeElement(this.getChannel(i));
        }
    }
}
```

Quadro 14 - Método implementado para envio de mensagens

3.3.2.2 Módulo cliente

Ao iniciar a aplicação servidora, inicia-se uma busca por dispositivos com *Bluetooth* ativo. Essa busca é realizada através da classe `InquiryMenu`. Cada dispositivo encontrado é feito a verificação se o mesmo tem o endereço MAC do servidor. Este endereço é informado de forma estática, conforme o dispositivo *Bluetooth* usado para testes.

Ao encontrar o servidor, é montado uma lista, através do método `deviceDiscovered()` (Quadro 15), com os três serviços disponíveis ao usuário (Pedido, Promoção e Enquete).

```
public void deviceDiscovered(RemoteDevice device, DeviceClass deviceClass) {
    this.device = device;
    //Aqui é feita a verificação se é o servidor
    if(this.device.getBluetoothAddress().equals("00158307CE2E")){
        this.append("Iniciar pedido", null);
        this.append("Promoções", null);
        this.append("Enquetes", null);

        if (this.select == null) {
            this.select = new Command("Iniciar", Command.OK, 1);
            this.addCommand(select);
        }
        this.setTitle("Servidor encontrado");
        //Como o servidor já foi encontrado, a busca é cancelada
        this.deviceDiscoverer.cancelInquiry();
        this.removeCommand(stopOrBack);
        this.stopOrBack = new Command("Voltar", Command.BACK, 1);
        this.addCommand(this.stopOrBack);}}}
```

Quadro 15 - Método invocado ao encontrar algum dispositivo com *Bluetooth* ativo

¹¹ *Broadcast*: é o processo pelo qual se transmite ou difunde determinada informação, tendo como principal característica que a mesma informação está sendo enviada para muitos receptores ao mesmo tempo

Com a conexão estabelecida, a aplicação chama o método `connectionEstablished()` (Quadro 16) da classe `Apresentacao`, e conforme a solicitação do usuário envia uma requisição ao servidor.

```

public void connectionEstablished(ClientDevice device) {
    if (!this.cancelled) {
        device.startListening();
        this.waitMessage.setDevice(device);
        BluetoothMenu.instance.setCurrent(this.waitMessage);
        try {
            if(itemSelecionado == 0){//Pedido
waitMessage.sendMessage("#"+device.getBluetoothAddress() + "#cSOLCARDAPPIO");
            }else
                if(itemSelecionado == 1){//Promoção
waitMessage.sendMessage("#"+device.getBluetoothAddress() + "#cPROMOCAO");
                }else{
                    if(itemSelecionado == 2){//Enquete
waitMessage.sendMessage("#"+device.getBluetoothAddress() + "#cPESQUISA");
                    }
                }
        } catch (BluetoothStateException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}

```

Quadro 16 - Método implementado para executar após a conexão estabelecida

Na classe `waitMessage` é onde todas as mensagens enviadas do servidor são recebidas e tratadas através do método `receiveMessage()` (Quadro 17).

```

01 public void receiveMessage(byte[] message) {
02     setTitle("Mesagem recebida");
03     //atributo que confirma recebimento do pedido
04     pedidoRecebido = false;
05     this.messageRec = new String(message);
06     //Verifica se a mensagem é destinada ao dispositivo que a está recebendo
07     if(isToken(getBluetoothLocal())){
08         //Verifica a requisição enviada pelo servidor
09         setTipoRequisicao();
10     //Inicia telas conforme tipo da requisição
11         iniciaRequisicao();
12     }
13 }

```

Quadro 17 - Método que monitora o recebimento de mensagens vindas do servidor

Dentro do método `receiveMessage()` (linha 11), há a chamada do método `iniciaRequisicao()` (), onde há o tratamento de cada requisição em particular. O método de envio de mensagem é implementado da mesma forma que no servidor, conforme já mostrado no Quadro 14. As mensagens são enviadas quando o usuário finaliza o pedido, fecha a promoção (somente envia a mensagem da promoção se o usuário confirmar a leitura) ou responde a enquete.

```

private void iniciaRequisicao(){
//Servidor rejeita a requisição do cliente por ele não ser cadastrado e envia
//uma mensagem informando a situação
    if(tipoRequisicao.equals("cUSUNCADASTRADO")){
        Saida saida = new Saida("Usuário não cadastrado",false,this);
        BluetoothMenu.instance.setCurrent(saida);
    }else{
        //Ponto de atendimento e cardápio recebido do servidor. Aqui o
        //cliente monta as telas de pedido
        if(tipoRequisicao.equals("cPEDIDO")){
            mensagemPedido = new MensagemPedido(this.messageRec,this.mac);
            chamaPontoAtend();
        }else{
            //Cliente monta a tela com a primeira promoção recebida do servidor
            if(tipoRequisicao.equals("cPROMOCAO")){
                this.mensagemPromocao =
                    new MensagemPromocao(this.messageRec,this.mac,false);
                chamaPromocao();
            }else{
                //Cliente monta tela informando que não há promoções para este usuário
                if(tipoRequisicao.equals("cNPROMOCAO")){
                    this.mensagemPromocao =
                        new MensagemPromocao(this.messageRec,this.mac,true);
                    chamaPromocao();
                }else{
                    //Cliente monta a tela com a primeira enquete recebida do servidor
                    if(tipoRequisicao.equals("cPESQUISA")){
                        this.menPesq =
                            new MensagemPesquisa(this.messageRec,getBluetoothLocal(),false);
                        chamaPesquisa(false);
                    }else{
                        //Cliente monta tela informando que não há enquetes para este usuário
                        if(tipoRequisicao.equals("cNPESQUISA")){
                            this.menPesq =
                                new MensagemPesquisa(this.messageRec,getBluetoothLocal(),true);
                                chamaPesquisa(true);
                        }else{
                            //Mensagem enviada do servidor informando que o pedido
                            //foi registrado com sucesso
                            if(tipoRequisicao.equals("cPEDIDORECEBIDO")){
                                pedidoRecebido = true;
                            }
                        }
                    }
                }
            }
        }
    }
}

```

Quadro 18 - Método que chama telas conforme requisição

3.3.3 Operacionalidade da implementação

A operacionalidade da implementação pode ser descrita dependendo da forma que se tem a intenção de utilizá-la. Podem ser citados alguns exemplos, como mostra o Quadro 19.

Possíveis utilizações da implementação			
Onde	Pedido	Enquete	Promoção
Cinemas	solicitar produtos como refrigerante ou pipoca da poltrona onde está sentado, evitando filas.	receber pesquisa de filmes ou gêneros preferidos, possibilitando assim, o desenvolvimento de campanhas para cliente específicos	receber promoção referente ao filme ou gênero favorito do cliente, atingindo assim um público específico
Estádio de futebol		Receber perguntas sobre seu time e acumular pontos ou mesmo a preferência por lugares	Receber promoção de ingressos ou brindes referentes ao gosto do torcedor
Bares ou lanchonetes	fazer pedidos da mesa onde está, pois em lugares cheios, a demora torna-se inconveniente	receber enquetes referente ao produtos e o atendimento, somando pontos para possíveis descontos	relacionando os pedidos dos clientes, há a possibilidade de criar campanhas usando esses dados

Quadro 19 - Possíveis utilizações da implementação

Este trabalho tem por objetivo demonstrar a aplicação em uma lanchonete. Cadastros e cenários foram elaborados com foco neste ambiente.

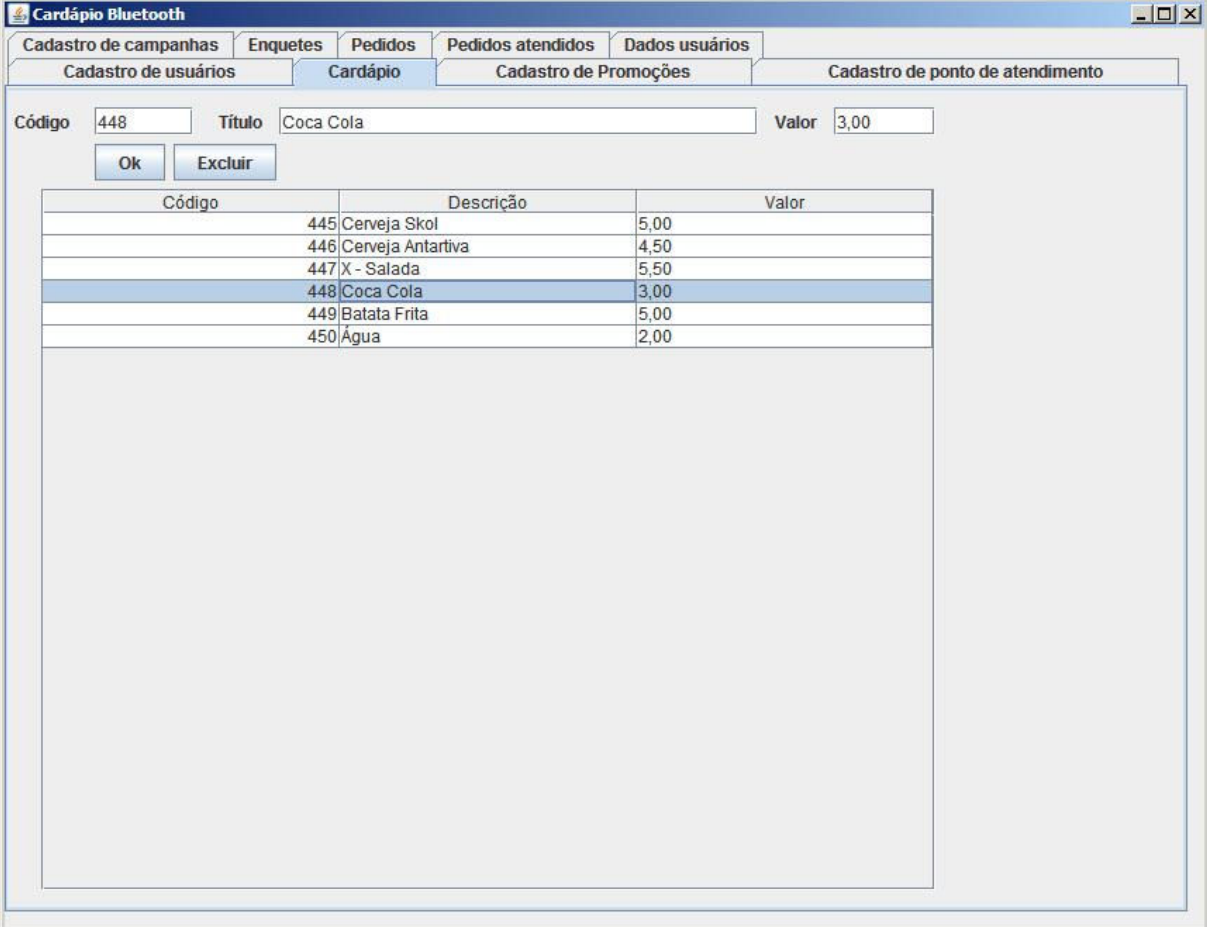
Ao executar a aplicação, a primeira tela apresentada é a tela de cadastro de usuários, conforme mostra a Figura 22. Nesta tela há o botão Procurar dispositivos, onde o usuário tem a facilidade em encontrar os dispositivos que estão a volta, para descobrir o endereço MAC e o apelido do dispositivo. Lembrando que não é necessário a busca por dispositivos para efetuar o cadastro. Abaixo encontra-se os campo para cadastro e uma tabela contendo todos os usuários cadastrados. Para alterar o cadastro de um usuário, selecione na tabela o usuário de deseja alterar, faça a alteração e clique no botão salvar. Para excluir, selecione o usuário na tabela e clique no botão excluir.

Código	MAC	Apelido	Nome	Sobrenome
1	0021FE738301	Rafael cel	Rafael	Formento
2	0017D57C130B	Gilberto Vogel	Gilberto	Vogel
3	0012D238EC7E	andre-cel	André	Formento
4	001B59F1D6E6	Jean	Jean	Bastos
5	00265D387B30	Samsung C3050	Ivan	Wilhelm

Figura 22 - Tela do cadastro de cliente

Na tela do cadastro do cardápio, mostrado na Figura 23, é possível cadastrar itens que serão enviados ao dispositivo que solicitou fazer um pedido. É necessário cadastrar o código, a descrição e o valor de cada item. Para alterar ou excluir cada item, basta clicar na tabela, então o item selecionado é carregado nos campos correspondentes e aí sim, o usuário tem condições de manipular os dados, clicando no botão salvar ou excluir cada registro.

Os cadastros de Promoções e Ponto de atendimento se fazem da mesma forma. Nas telas há campos para cadastro, tabela com dados já cadastrados e os botões para manipulação dos dados.



Código	Descrição	Valor
445	Cerveja Skol	5,00
446	Cerveja Antartiva	4,50
447	X - Salada	5,50
448	Coca Cola	3,00
449	Batata Frita	5,00
450	Água	2,00

Figura 23 - Tela de cadastro de cardápio

A Figura 24 mostra a tela de escolha de opções, que servirá a todos os serviços no módulo cliente. Depois de cadastrado os itens essenciais, como os pontos de atendimento e o cardápio, o usuário do dispositivo móvel pode começar a fazer os pedidos.



Figura 24 - Tela de escolha do serviço no cliente

Escolhida a opção serviço, há três passos para finalizar um pedido. O primeiro é a escolha do ponto de atendimento, depois a escolha dos itens e as quantidades e por fim a finalização, com envio ao servidor e a confirmação da mensagem enviada. Os três passos podem ser analisados observando a Figura 25.



Figura 25 - Três passos para efetuar um pedido no módulo cliente

No momento em que a aplicação servidor recebe um pedido, esta lista o pedido em uma tela, informando ainda cada item do pedido recebido (Figura 26).

The screenshot shows a software interface titled 'Cardápio Bluetooth'. It features a menu bar with tabs: 'Cadastro de campanhas', 'Enquetes', 'Pedidos', 'Pedidos atendidos', and 'Dados usuários'. Below the menu bar are sub-tabs: 'Cadastro de usuários', 'Cardápio', 'Cadastro de Promoções', and 'Cadastro de ponto de atendimento'. The main content area is divided into two sections: 'Pedidos' and 'Itens'.

Pedidos

Nº Pedido	Mesa	Usuário	Total	Solicita garçom	Atendido
41	Mesa 1	Rafael	5	<input checked="" type="checkbox"/>	<input type="checkbox"/>
42	Mesa 1	Rafael	3	<input type="checkbox"/>	<input type="checkbox"/>
43	Mesa 1	Rafael	3	<input type="checkbox"/>	<input type="checkbox"/>
44	Mesa 1	Rafael	3	<input type="checkbox"/>	<input type="checkbox"/>
45	Mesa 1	Rafael	6	<input type="checkbox"/>	<input type="checkbox"/>
46	Mesa 2	Rafael	11	<input type="checkbox"/>	<input type="checkbox"/>
47	Mesa 2	Rafael	11	<input type="checkbox"/>	<input type="checkbox"/>
48	Mesa 2	Rafael	11	<input type="checkbox"/>	<input type="checkbox"/>
49	Mesa 2	Rafael	3	<input checked="" type="checkbox"/>	<input type="checkbox"/>
50	Mesa 1	Rafael	11	<input type="checkbox"/>	<input type="checkbox"/>
51	Mesa 2	Rafael	11	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Itens

Código produto	Descricao	Quantidade	Valor
448	Coca Cola	2.0	3
449	Batata Frita	1.0	5

At the bottom right of the interface is a button labeled 'Fechar pedido'.

Figura 26 - Tela com informação dos pedidos recebidos

No cadastro de campanhas (Figura 27), o usuário informa um código, título e se deseja repetir a campanha depois que um usuário já a recebeu. Abaixo o usuário seleciona a promoção referente a campanha e os usuários que a receberão. Na tabela de usuários há uma coluna informando se o usuário já recebeu a campanha.

Nesta tela existem três botões: o primeiro, o botão limpar, retira os itens selecionados nas tabelas de promoção e usuários; o botão salvar, que guarda as informações em disco e o botão excluir, que exclui o registro referente ao item que está selecionado na tabela de campanhas cadastradas.

Cardápio Bluetooth

Cadastro de campanhas Enquetes Pedidos Pedidos atendidos Dados usuários

Cadastro de usuários Cardápio Cadastro de Promoções Cadastro de ponto de atendimento

Código Título Repetir

Promoções

Código	Nome
1	Inauguração
2	Promoção para batata frita

Usuarios

MAC	Nome	Sobrenome	Enviado
0021FE738301	Rafael	Formento	<input checked="" type="checkbox"/>
0017D57C130B	Gilberto	Vogel	<input type="checkbox"/>
0012D238EC7E	André	Formento	<input type="checkbox"/>
001B59F1D6E6	Jean	Bastos	<input type="checkbox"/>
00265D387B30	Ivan	Wilhelm	<input type="checkbox"/>

Campanhas cadastradas

Código	Campanha	Repetir
1	Inauguracao	<input checked="" type="checkbox"/>
2	Cientes que pedem batata	<input type="checkbox"/>

Figura 27 - Tela de cadastro de campanhas

No módulo cliente, cada usuário pode receber mais de uma promoção. Cada promoção recebida o usuário tem a opção de confirmar a leitura. Essa confirmação faz somar uma pontuação para cada usuário no servidor. Com isso, o usuário é obrigado a passar por toda promoção para chegar ao ponto de confirmar a leitura (Figura 28).



Figura 28 - Telas de campanhas no módulo cliente

Para cadastrar uma enquete, o usuário seleciona a aba Enquetes. Nesta tela há campos para cadastro, como o código, multi-seleção e os campos das opções de resposta. A opção multi-seleção possibilita o usuário a escolher mais de uma resposta, enquanto a seleção única possibilita apenas uma resposta. Para acompanhar uma enquete, com a quantidade de respostas e a porcentagem de cada resposta, ao lado de cada opção há uma indicação de porcentagem. A tela do cadastro de enquetes pode ser visualizada na Figura 29.

Cardápio Bluetooth

Cadastro de campanhas Enquetes Pedidos Pedidos atendidos Dados usuários

Cadastro de usuários Cardápio Cadastro de Promoções Cadastro de ponto de atendimento

Enquetes

Código	Título
2	O que gosta de comer no final de semana
1	Qual forma de pagamento de sua preferência

Código Título Multi-seleção

Opção 1

Opção 2

Opção 3

Usuários Total de participações: 2

MAC	Nome	Sobrenome	Enviado
0021FE738301	Rafael	Formento	<input checked="" type="checkbox"/>
0017D57C130B	Gilberto	Vogel	<input type="checkbox"/>
0012D238EC7E	André	Formento	<input type="checkbox"/>
001B59F1D6E6	Jean	Bastos	<input type="checkbox"/>
00265D387B30	Ivan	Wilhelm	<input type="checkbox"/>

Figura 29 - Tela de cadastro de enquetes

A Figura 30 demonstra duas enquetes sendo recebidas por um cliente.



Figura 30 - Telas de enquetes no módulo cliente

Para fins de acompanhamento do cliente, com possibilidade de verificar o uso da aplicação, no servidor há uma tela que é possível visualizar o fluxo de interação de cada cliente com a aplicação. A tela Dados Usuários mostra informações como a quantidade de pedidos, campanhas ou enquetes que cada usuário participou. Ainda mostra uma tabela com os itens mais pedidos, podendo assim influenciar em uma campanha ou enquete para determinados usuários (Figura 31).

The screenshot shows a web application window titled 'Cardápio Bluetooth'. It has several tabs: 'Cadastro de campanhas', 'Enquetes', 'Pedidos', 'Pedidos atendidos', and 'Dados usuários'. The 'Dados usuários' tab is active, showing a table of user information and summary statistics.

Código	MAC	Apelido	Nome	Sobrenome
1	0021FE738301	Rafael cel	Rafael	Formento
2	0017D57C130B	Gilberto Vogel	Gilberto	Vogel
3	0012D238EC7E	andre-cel	André	Formento
4	001B59F1D6E6	Jean	Jean	Bastos
5	00265D387B30	Samsung C3050	Ivan	Wilhelm

Summary statistics:

- Pedidos** 44
- Campanhas** 2
- Enquetes** 2

Itens mais pedidos pelo cliente:

Código	Descrição	Quantidade
448	Coca Cola	32.0
445	Cerveja Skol	12.0
449	Batata Frita	9.0
447	X - Salada	4.0
450	Água	3.0
446	Cerveja Antartiva	2.0

Figura 31 - Tela fluxo de dados de cada usuário

3.4 RESULTADOS E DISCUSSÃO

Os resultados encontrados com o término do trabalho são satisfatórios, pois com a utilização da aplicação é possível fazer uma comunicação *Bluetooth*, com envio de mensagens personalizadas à cada dispositivo, portanto alcançando o objetivo inicial.

No Quadro 20 é apresentado o comparativo das características da aplicação com os trabalhos correlatos. Pode-se observar que todas as aplicações são desenvolvidas com J2ME,

porém usam comunicações diferentes. Enquanto apenas este projeto e o projeto de Sedrez (2006) utilizam um framework, que facilita o desenvolvimento da aplicação.

Apesar deste projeto ser voltado para o comércio móvel, há diferenças em relação ao projeto de Sedrez (2006), como o fato de que este projeto atende um público mais próximo, pela limitação do *Bluetooth*, enquanto Sedrez (2006), que usa *Web Services*, tem um alcance maior ao cliente.

A aplicação não contempla a implementação de uma conexão segura. Esta seria a próxima etapa para tornar a aplicação mais próxima do ideal para comercialização. No quesito da aplicação ser mais voltada ao comércio móvel, considera-se um cenário voltado ao marketing, que desta forma torna esta aplicação a mais indicada.

Outro fato interessante, é a característica de nenhuma implementação armazenar dados no celular. Isto torna a aplicação mais ágil, pois não se perde tempo com acessos a arquivos de armazenamento de dados.

	Este projeto	SEDREZ	TRUPEL
Aplicação desenvolvida com J2ME	✓	✓	✓
Aplicação voltada para o comércio móvel	✓	✓	✗
Não necessita conexão com internet	✓	✗	✓
Conexão segura	✗	✓	✗
Utilização de framework	✓	✓	✗
Armazenamento das informações do celular	✗	✗	✗
Comunicação	Bluetooth	Web services	Socket

Quadro 20 – Características da aplicação e trabalhos correlatos

4 CONCLUSÕES

No início do desenvolvimento da aplicação foram elencadas funcionalidades que ele deveria ter como: iniciar uma conexão entre um celular e um computador, interar através de mensagens os dois dispositivos com uma conexão *Bluetooth*, para que a funcionalidade principal deste trabalho pudesse ser implementada, que é facilitar a interação com clientes que possuem dispositivos móveis através do *Bluetooth*. Ao término pode-se observar que as funcionalidades foram criadas, atingindo os objetivos.

Uma das dificuldades encontradas foi fazer a primeira transmissão via *Bluetooth*. Poucos exemplos e material escasso na área dificultaram o entendimento e o desenvolvimento da transmissão. Após muito tempo e muito material lido, foi constatado uma falha no dispositivo *Bluetooth*. O dispositivo não interagiu corretamente com a pilha de protocolos da Microsoft, sendo necessária a compra de um novo dispositivo. Depois disto, com algumas implementações em cima do framework Marge, conseguiu-se a primeira conexão.

Houve também a dificuldade em relação as mensagens enviadas entre cliente e servidor. *Layouts* foram montados para haver o entendimento em relação aos serviços e conteúdos transmitidos entre servidor e clientes.

O uso do framework Marge para o desenvolvimento da aplicação facilitou a implementação e o entendimento da mesma. Este framework abstrai ainda o uso da biblioteca Bluecove, que seria uma implementação complexa da especificação JSR-82.

Um estabelecimento que necessite de mais agilidade no atendimento ao seus clientes, como o exemplo de uma lanchonete, pode utilizar esta aplicação para atender mais rápido e com mais precisão. Além de contar com serviços de promoções personalizadas e também enquetes, que ajudam na melhora dos serviços e pedidos. Contando sempre com a privacidade das informações, assunto importante quando se trata de uma interação direta com o cliente. Todos estes benefícios na palma da mão do usuário.

No entanto, existem algumas limitações no uso da aplicação: quando cliente perde comunicação com o servidor ou vice-versa, a aplicação não trata o erro; não foi possível obter marca e modelo do celular por falta de documentação; quando ocorre um erro no servidor é necessário reiniciar a aplicação para retomar a conexão.

Contudo, este trabalho me deu uma nova visão do mobile marketing, mostrando o quanto ele está presente em nossas vidas e o quanto se pode trabalhar com este segmento. As oportunidades aparecem a cada dia, e a cada dia me identifico mais com esta área. Entendo

que esta pesquisa e desenvolvimento me incentivou a buscar mais conhecimento acadêmico e profissional.

4.1 EXTENSÕES

Existem pontos que podem ser agregados ou melhorados na aplicação. Como sugestão pode-se citar:

- a) adicionar novos serviços entre cliente e servidor, como o envio de perguntas ao servidor ou mesmo um *chat*;
- b) adicionar novas funcionalidades ao serviço de enquete, como criar perguntas automaticamente em relação aos produtos que cada usuário mais consumiu;
- c) implementar um processo de pontuação mais elaborado, criando assim promoções mais dinâmicas.

REFERÊNCIAS BIBLIOGRÁFICAS

AGÊNCIA BRASIL. **Número de assinantes de telefonia celular no Brasil já ultrapassa 151 milhões**. Brasília, 2009. Disponível em:

<<http://www.agenciabrasil.gov.br/noticias/2009/02/18/materia.2009-02-18.3752888527/view>>. Acesso em: 11 mar. 2009.

ALECRIM, Emerson. **Tecnologia Bluetooth**. 2008. Disponível em:

<<http://www.infowester.com/bluetooth.php>>. Acesso em: 26 mar. 2009.

ALENCAR, Marcelo Sampaio de. **Telefonia celular digital**. São Paulo: Érica, 2004.

GHSI, Bruno Cavaler. **Marge: framework** para desenvolvimento de de aplicações em Java que façam uso da tecnologia *Bluetooth*. 2007. 168 f. Trabalho de Conclusão de Curso (Bacharel em Sistemas de Informação) – Centro Tecnológico, da Universidade Federal de Santa Catarina, Florianópolis.

GOELZER, Maiquel. Bluecove: comunicando aplicativos J2SE com aplicativos J2ME através de *Bluetooth*. **WebMobile**, Rio de Janeiro, n. 08, p. 21-27, 2006.

HENRIQUE, Eduardo L.; O uso do *Bluetooth* como ferramenta de marketing. **WebMobile**, Rio de Janeiro, n.13, p. 24-26, 2007.

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

KOTLER, Philip. **Marketing para o século XXI**: como criar, conquistar e dominar mercados. Tradução Bazán Tecnologia e Lingüística. São Paulo: Futura, 1999.

LANGE, Talvani; SHROEDER, Rafael de Tarso. **Mídia celular, publicidade e consumo estratificado**. 2008. Disponível em: <http://www.eca.usp.br/pjbr/arquivos/dossie10_d.htm>. Acesso em: 26 mar. 2009.

LEE, Valentino; SCHNEIDER, Heather; SCHELL, Robbie. **Aplicações móveis**: arquitetura, projeto e desenvolvimento. Tradução Amaury Bentes & Deborah Rüdiger. São Paulo: Pearson Education do Brasil, 2005.

LIMEIRA, Tania Maria Vidigal. Fundamentos de marketing. In: DIAS, Sergio Roberto (Coord.). **Gestão de marketing**. São Paulo: Saraiva, 2006. p. 1-15.

LINS, Eduardo. **Bluetooth marketing já começou no Brasil**. 2006. Disponível em: <<http://webinsider.uol.com.br/index.php/2006/11/03/bluetooth-marketing-ja-comecou-no-brasil>>. Acesso em: 14 set. 2009.

Marge, Java *Bluetooth* Framework. Disponível em: <<https://marge.dev.java.net>>. Acesso em: 14 set. 2009.

MATTOS, Érico Tavares de. **Programação Java para wireless**: aprenda a desenvolver sistemas em J2ME. São Paulo: Digerati Books, 2005.

MCKENNA, Regis. **Acesso total**: o novo conceito de marketing de atendimento. Tradução Elaine Aparecida Pepe. Rio de Janeiro: Campus, 2002.

MILLER, Michael. **Descobrimo *Bluetooth***: aprenda o que você pode fazer com a tecnologia *Bluetooth* hoje – e o que poderá fazer amanhã. Tradução: Altair Dias Caldas de Moraes e Cláudio Belleza Dias. Rio de Janeiro: Campus, 2001.

MUCHOW, J. W. **Core J2ME tecnologia e MIDP**. Tradução João Eduardo Nóbrega Tortello. São Paulo: Pearson Makron Books, 2004.

POSTMA, Paul. **The new marketing era**. New York: McGraw-Hill, 1999.

PRADO, Eduardo. **Mobile marketing**: uma nova mídia digital. 2006. Disponível em: <<http://www.teleco.com.br/emdebate/eprado21.asp>>. Acesso em: 26 mar. 2009.

SANTANA, Everton. **Wireless sobre controle**: usando o padrão 802.15.4 no gerenciamento de *appliances* em rede - o padrão IEEE 802.15.4. 2009. Disponível em: <http://almerindo.devin.com.br/index.php?option=com_content&view=article&id=86%3Awireless-sobre-controle-usando-o-padrao-802154-no-gerenciamento-de-appliances-em-rede&catid=43%3Atrabalhos-de-alunos&Itemid=18&limitstart=1>. Acesso em: 15 jan. 2009.

ROMÁN, Fernando; GONZÁLEZ-MESONES, Fernando; MARINAS, Ignacio. **Mobile marketing**: a revolução multimídia. Tradução e revisão técnica Paco Torras. São Paulo: Thomson Learning, 2007.

RUFINO, Nelson Murilo de Oliveira. **Segurança em redes sem fio**: aprenda a proteger suas informações em ambientes Wi-Fi e *Bluetooth*. 2. ed. São Paulo: Novatec, 2007.

SEDREZ, Daiana Maria. **Aplicação comercial para celulares baseada em M-Commerce utilizando J2ME**. 2006. 99 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

SUN DEVELOPER NETWORK. **Java ME platform overview**. 2009. Disponível em: <<http://java.sun.com/javame/technology/index.jsp#convergedservices>>. Acesso em: 5 set 2009.

TORRI, Lucas; GHISI, Bruno. Marge: *Bluetooth* fácil para desktop e dispositivos móveis. **Mundo Java**, Curitiba, n. 30, p. 32-41, jul, 2008.

TRUPEL, Wilson. **Desenvolvimento de um jogo multiplayer para celular**. 2008. 69 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

TUDE, Eduardo. *Bluetooth*. 2004. Disponível em:
<<http://www.teleco.com.br/tutoriais/tutorialblue/default.asp>>. Acesso em: 12 set. 2009.