Centro Universitário de Brasília.

Faculdade de Ciências Exatas e Tecnologia.

Curso de Engenharia da Computação

ANDRÉ RABELO GRAVINA

SISTEMA DE COMPRA EM MÁQUINAS DE AUTO-ATENDIMENTO UTILIZANDO APARELHO CELULAR

BRASÍLIA, 1º SEMESTRE DE 2006.

ANDRÉ RABELO GRAVINA
RA: 2001563/0

SISTEMA DE COMPRA EM MÁQUINAS DE AUTO-ATENDIMENTO UTILIZANDO APARELHO CELULAR

Trabalho apresentado ao Centro Universitário de Brasília (UNICEUB) Como pré-requisito para a obtenção de Certificado de Conclusão do Curso de Engenharia da Computação.

Orientador: M.C. Professor Francisco Javier de Obaldia Diaz

BRASÍLIA, 1º SEMESTRE DE 2006.

DEDICATÓRIA

À Deus, à minha abençoada família e à minha namorada, pelo inesgotável apoio e torcida.

AGRADECIMENTOS

Agradeço primeiramente a Deus pelas oportunidades de aprendizado e crescimento que me foram oferecidas e pelo direcionamento para realização de minhas tarefas.

Aos meus virtuosos pais, Carlos Roberto Gravina e Regina Rabelo, por estarem presentes em todas as etapas da minha vida, me apoiando e incentivando.

Às minhas irmãs Roberta, Carolina e Daniela e ao meu irmão Eduardo pela forte união que tanto nos fortalece.

À minha namorada Laila Vieira com quem pude compartilhar momentos de alegria e dificuldade ao longo deste projeto.

Aos meus amigos, em especial ao Analista de Sistemas Leonardo Pereira Carvalho e aos Engenheiros João Paulo Galvagni Junior e Luciano Duarte Nascimento Franco pela valiosa dedicação e orientações técnicas.

Por fim, agradeço ao meu orientador Javier por toda dedicação e auxilio para a concretização deste projeto.

RESUMO

Aproveitando o grande desenvolvimento dos negócios por intermédio de telefones móveis (m-business), este projeto de graduação tem como propósito apresentar um sistema que possibilite a realização de compras em máquina de auto-atendimento utilizando o aparelho celular.

A realização da compra é feita utilizando o envio e recebimento de mensagens SMS, que apesar de ser encontrado em diversas tecnologias, será trabalho em conjunto com celulares de tecnologia GSM.

Neste projeto foram desenvolvidos mecanismos capazes de simular o tramite entre o cliente que realiza a compra, a operadora de telefonia móvel que disponibiliza o serviço de compra diferenciado e a máquina de auto-atendimento.

Palavras-Chaves: m-business, GSM, auto-atendimento.

ABSTRACT

Because of the pervasive of mobile phones in business, this graduation project proposes to show a system that makes shopping with an automated machine, using a cell phone, possible.

Purchases are made by using cell phones with GSM technology which send and receive SMS messages.

Mechanisms capale of simulating the transmissions between the buyer and the mobile tephone operator were developed to create a new type of business.

Keywords: m-business, GSM, Automated Machines.

SUMÁRIO

LISTA DE SÍMBOLOS

API – Application Programming Interface.

ARPANET - Advanced Research Projects Agency - Administração de Projetos de Pesquisa Avançados

AUC - Authentication Center.

BSS - Base Station.

DTD – Document Type Definition.

ETSI - European Telecommunication Standards Institute.

E/S – Entrada e Saída.

GPRS – General Packet Radio Service.

GSM – Global System for Mobile Communication.

HLR -Home Location Register.

HTTP – Hyper Text Transport Protocol.

IMEI - International Mobile Equipment Identity (Identificação Internacional de Equipamento Móvel).

JSP - Java Server Pages.

MS - Móbile Station.

MSC - Móbile-Services Switches Centre.

M2M – Máquina e Máquina.

PDA – Personal Digital Assistant.

SMS - Short message service ou Serviço de mensagens curtas

TCP/IP Transmission Control Protocol/Internet Protocol

VLR - Visitor Location Register.

XML - Extensible Markup Language.

WARC - World Administrative Radio Conference.

WAC - WIG Application Creator.

WIG - Wireless Internet Gateway.

WML – Wireless Markup Language.

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

ÍNDICE DE QUADROS

CAPÍTULO 1

INTRODUÇÃO

1.1. MOTIVAÇÃO

Paralelamente ao avanço da tecnologia, existe uma crescente necessidade de consolidação de funcionalidades e acesso a aplicações antes separadas, e agora, passíveis de serem utilizadas somente de maneira remota.

O telefone celular, um fenômeno de aceitação por parte da população, com crescimento vertiginoso em praticamente todas as regiões do globo, se constitui no principal aparelho móvel utilizado, pelo menos em relação ao número de usuários.

Visando este enorme mercado, as empresas de telecomunicação investiram em alta tecnologia em busca de serviços diferenciados para atrair novos clientes.

Esse impulso tecnológico foi responsável por grandes alterações no cenário de telefonia móvel. Aos poucos, a transferência de voz torna-se apenas uma opção básica entre tantas outras possibilidades e, ainda, entre uma grande variedade de serviços digitais.

A convergência de aplicações, assim como das funcionalidades, em um único aparelho é uma tendência que vai de encontro, pelo menos na teoria, da simplificação do uso, assim como do aumento da portabilidade, ou seja, menos aparelhos a se carregar.

Aproveitando esta nova tendência, este projeto propõe inserir uma nova funcionalidade ao aparelho celular. Seria uma evolução do cartão de crédito, onde o valor das compras será debitado na conta telefônica do cliente.

Essa nova funcionalidade será implementada neste projeto para o caso particular de compras em máquinas de auto-atendimento como refrigerantes, salgadinhos, café, etc.

1.2. OBJETIVO

Este projeto tem como objetivo desenvolver um sistema de compra em máquinas de auto-atendimento utilizando os recursos do aparelho celular para efetuar o pagamento, unindo o conforto e praticidade que a tecnologia pode oferecer a um consumidor exigente. Trata-se de uma idéia de evolução nas formas de pagamento que passaram por moeda, cédula, cheque, cartão de crédito e agora transações via telefone celular.

De forma simplificada o processo decorrerá da seguinte maneira:

- O Cliente adquire o serviço junto à sua operadora de telefonia para realizar compras que sejam debitadas em sua conta telefônica.
- Diante da máquina de auto-atendimento o usuário deste serviço enviará uma mensagem SMS (Short message service ou Serviço de mensagens curtas) para um servidor da operadora de telecomunicação contendo informações sobre a compra que será realizada. Essas informações serão: qual é o produto a ser comprado, qual é a máquina fornecedora, e informação sobre o cliente.
- Essas informações serão validadas no servidor assim como a adimplência do cliente junto sua operadora. O valor da compra é então debitado na fatura de telefone do cliente.
- Após a validação será enviado via telefone celular, um código ao cliente que deve ser digitado na máquina de auto-atendimento para liberação do produto para consumo.

Este serviço de compra via celular trará benefícios tais como:

Para o consumidor:

- Efetuar compras mesmo sem possuir em mãos a quantidade necessária de moedas ou cédulas de dinheiro;

- O valor da compra será debitado na fatura telefônica do cliente ou abatido no crédito de seu celular pré-pago.

Para os proprietários das máquinas de auto-atendimento:

- Possível aumento das vendas proporcionada pela facilidade, comodidade e conveniência disponibilizada ao cliente;
- Não haverá necessidade de recolhimento do dinheiro armazenado nas máquinas, o que diminui custos e evita problemas com roubos e assaltos.
- Possibilidade de controle on-line de estoque e faturamento. Ou seja, essas informações poderão ser repassadas pela operadora ao fornecedor da máquina de auto-serviço via Internet, por exemplo, em tempo real.

Para as operadoras de telefonia celular é um serviço diferenciado que pode contribuir para atrair novos clientes e ainda manter os existentes.

Para aplicação prática (comercial) do projeto, seria necessária uma parceira entre a empresa de telecomunicações e a fornecedora de máquinas de auto-atendimento. Isso porque a empresa de telefonia deve manter todo o controle das cadastro máquinas de auto-atendimento e seus produtos, controle de estoque e vendas realizadas. Para então, repassar os valores dos produtos vendidos aos fornecedores das máquinas.

Também é necessária a adaptação das diferentes máquinas de auto-serviço para este projeto.

Devido a essas limitações, todo o trâmite comercial será sintetizado neste projeto de forma acadêmica, ou seja, por intermédio de simulações via software.

Será criado um protótipo capaz de simular equipamentos e tecnologias que por hora não temos acesso, como, por exemplo, servidor de mensagens da operadora de telefonia móvel e a máquina de auto-atendimento.

Cabe salientar que a segurança dos dados trafegados entre os sistemas, assim como o controle da geração dos códigos de autorização de venda/compra, não constam no escopo deste projeto. Para uma aplicação real seria necessário que esse código fosse alterado periodicamente para evitar problemas com fraudes.

As simulações por software desconsideram estas questões, que serão citadas como sugestões para trabalhos futuros no capitulo 5.

Para um momento futuro, existe o propósito de inserir esta idéia no mercado, a médio prazo, trazendo benefícios e comodidades aos clientes (usuários), às empresas de telecomunicação, dinamizando o comércio.

1.3. ESTRUTURA DA MONOGRAFIA

Esta monografia está estruturada em cinco capítulos de acordo com as fases de desenvolvimento do projeto. Após a introdução decorrida neste primeiro capítulo este trabalho de graduação está segmentado conforme o detalhamento a seguir.

No Capítulo II estará todo o embasamento teórico do projeto. Serão tratados pontos como a influência do aparelho celular nas transações comerciais e um detalhamento das tecnologias utilizadas.

O Capítulo III abrangerá as topologias, as soluções em software e hardware baseado nas tecnologias descritas e detalhadas no Capítulo II.

O Capítulo IV será o de resultados: telas com interfaces, demonstrativos gráficos, demonstração da aplicação, itens de controle testados e seus resultados.

O Capítulo V trará as conclusões, novas sugestões de projetos e análise dos resultados.

CAPÍTULO 2

REFERENCIAL TEÓRICO – A INTERNET, A TECNOLOGIA CELULAR E SUAS APLICAÇÔES NO COMÉRCIO ELETRÔNICO.

Comunicar-se é uma necessidade do ser humano, podendo-se dizer que faz parte de sua própria natureza sendo o que o difere dos demais animais, pois além do som emitido, o mesmo possui uma ordem lógica devido a sua capacidade de raciocínio. Tendo esta necessidade, o ser humano em sua busca contínua de aperfeiçoamento sempre procurou alternativas de como se comunicar com seus semelhantes. Desde o uso de técnicas primitivas como sinais de fumaça, pombos correios e mensageiros, passando pelo uso da imprensa, até os modernos sistemas de comunicação existentes, impulsionados pelo uso do computador, sistemas de processamento e meios de comunicações que utilizam pares metálicos, fibras ópticas, comunicação sem fios, com modernos sistemas de comunicação satélite e sistemas de comunicação que utilizam as redes de celulares, destacando que qualquer um desses sistemas, permite hoje, o acesso a diversos tipos de serviços através da Internet.

2.1. A INTERNET

Podemos descrever a Internet com uma palavra: “comunicação”. Isso explica porque tanto o computador como a Internet estão cada vez mais presentes em nossas vidas. O ser humano necessita de informação para poder se comunicar com mais precisão, e essa comunicação é feita através de computadores conectados em redes.

Em 1969, quatro universidades americanas formam a ARPANET, uma rede de laboratórios que fazia estudos para a Administração de Projetos de Pesquisa Avançados (Advanced Research Projects Agency) do Departamento de Defesa dos Estados Unidos. Mais tarde, a Arpanet se transformaria na Internet, com o objetivo de descentralizar a transmissão de dados, evitando-se que as comunicações fossem cortadas durante um eventual ataque nuclear contra os Estados Unidos, portanto, a Internet é fruto da Guerra Fria [LAZILHA, 1999].

Após uma enorme expansão da Internet na década de 70, os anos 80 foram marcados pela introdução dos protocolos TCP/IP (Transmission Control Protocol/Internet Protocol), com normas técnicas para transmissão de informações através da rede e pela liberação pelos EUA, para uso comercial.

A grande explosão ocorreu a partir de 1993 com o surgimento da *World Wide Web – WWW* e a criação do programa *Mosaic*, primeiro browser para navegação na Internet, integrando textos e gráficos Web, popularizando de vez a rede mundial.

Desde então o avanço da Internet mostra uma evolução sem precedentes na historia das comunicações. Adotada por Instituições de pesquisas e Universidades e desenvolvidas por grandes empresas privadas a Internet demonstrou um enorme avanço quanto à capacidade de comunicação e facilidade de acesso e navegação

No ano de 1997, a utilização da rede explode em todo o mundo. O número de usuários ultrapassa a casa dos 50 milhões, sendo mais de um milhão no Brasil. Empresas de tecnologia iniciam a fase do comércio on-line.

A partir de 1997 surgem, então, as primeiras ofertas de vendas de produtos e serviços através da Internet, que, a partir de então, vêm crescendo de forma surpreendente.

De acordo com Paulo de Alencar, em matéria da revista INFO [INFO, 2006] , o número de internautas em todo o mundo chegou a um bilhão. O IBOPE [IBOPE, 2006] estima em 32,1 milhões o número de usuários de Internet no Brasil.

Esta significativa massa de usuários acessa a rede através de computadores e terminais em instituições educacionais, provedores comerciais e outras organizações. Além dos recursos básicos de correio eletrônico e lista de discussão, a Internet proporciona a seus usuários acesso aos mais variados serviços de informação como, por exemplo, bases de dados especializadas, catálogos de bibliotecas, repositórios de software de domínio público, jornais e revistas eletrônicas, etc. Através da Internet, também é possível ter acesso a recursos de hardware especializados como computadores de alto desempenho e processadores especializados.

2.1.1. A Internet no Brasil

O uso público irrestrito e comercial da Internet iniciou-se em 1995. A partir de abril de 1995, o Ministério das Comunicações e o Ministério da Ciência e Tecnologia decidiram lançar um esforço comum de implantação de uma rede integrada entre instituições acadêmicas e comerciais. A partir desse momento, vários fornecedores de acesso e serviços privados começaram a operar no Brasil [RNP, 1997].

Trezentos mil computadores estavam conectados à rede em 1996/1997, passando para um milhão em 1997/1998, atingindo 1,8 milhão no primeiro semestre de 1999. Atualmente temos o seguinte panorama da Internet no Brasil:

Quadro 2.1 – Usuários de Internet no Brasil (IBOPE Net Ratings 2006)

	Usuários 2005		**Usuários 2006**	
	Ativos	**Com acesso**	**Ativos**	**Com acesso**
Janeiro	10.656.901	17.945.437	12.035.681	21.227.222
Fevereiro	11.032.316		13.240.648	
Março	11.030.724		14.106.651	
Abril	11.378.029	18.336.044	13.431.424	21.227.222
Maio	11.517.361			
Junho	11.548.170			
Julho	11.434.547	18.336.044		
Agosto	11.630.195			
Setembro	11.960.385			
Outubro	11.729.619	18.892.455		
Novembro	12.529.892			
Dezembro	12.208.375			

O quadro 2.1 evidencia o rápido crescimento da Internet relacioando a quantidade de usuários ativos (cadastrados) e quantos acessam efetivamente a Internet exibidos na coluna “com acesso”.

Segundo a UIT em 2004 o Brasil era o 10º em número de usuários de Internet, o 7 º em número de Hosts (servidores) e o 19º em número de PCs no mundo, como pode ser demostrado na tabela 2.2 .

Quadro 2.2 – Usuários de Internet no Mundo. Fonte: UIT – International Telecommunication Union (World Telecommunication Indicators 2004).

	Milhões	Internet		PCs	Hosts
		Usuários	/100 hab		
1	US	185	62,3	220	195
1	China	94	7,2	53	0,16
3	Japão	75	58,7	69	16
4	Alemanha	41	50,0	46	3,0
5	Reino Unido	38	63,3	36	2,1
6	India	35	3,2	13	0,14
7	Coréia	31	65,7	26	5
8	Itália	29	50,3	18	1,6
9	França	25	41,4	29	2,3
10	Brasil	22	12,2	19	3,5

2.1.2. A Utilização da Internet pelas Empresas

A partir da liberação do uso da Rede Internet para fins comerciais, inúmeros serviços são oferecidos através da rede. Um dos melhores exemplos dessa utilização está no crescimento de sites voltados para o segmento de viagens e turismo, que permitem a obtenção de informações detalhadas sobre locais de destino, bem como a compra de passagens aéreas, reservas de hotéis e carros e a compra de ingressos para espetáculos. A rede também vem sendo utilizada para a realização dos seguintes serviços: assinaturas de revistas, compra e venda de imóveis, acesso a serviços de informações e banco de dados, edição de revistas eletrônicas, jornais on line, consultoria, home-banking, treinamento de redes e franquias, concessão de crédito, educação, realização de leilões, recrutamento de mão-de-obra e pregões eletrônicos de bolsas de valores [ALMEIDA, 1999].

Mais recentemente, vem sendo utilizada para a comercialização de produtos, porque as empresas estão buscando formas de chegar ao novo consumidor – e-customer, que demanda maior rapidez e eficiência com menor consumo de tempo.

As páginas das empresas na Internet voltam-se cada vez mais para os negócios e tornam-se, no mínimo, ferramenta de marketing para ampliar o contato com os clientes. Atualmente, fabricantes e fornecedores dos mais variados bens e serviços utilizam a Internet para divulgar seus produtos.

As empresas que atuam com o comércio de produtos, através da Internet, fazem-no com seus próprios Web sites, de forma independente ou juntando-se em espaços comerciais on-line para onde convergem os compradores da Internet.

Quanto à efetivação da compra, o consumidor faz o pedido através de formulário eletrônico do site ou do shopping virtual, referindo-se ao produto, quantidade, forma de pagamento e, ainda, indicando o local e hora mais conveniente para a entrega.

Embora a Internet venha funcionando muito mais como um canal de mídia do que de vendas, o comércio na Internet tem alcançado valores expressivos.

No Brasil , o VOL®, que representa a soma dos volumes de transações de automóveis, turismo e bens de consumo (lojas virtuais e leilões para pessoa física), atingiu, em 2005, R$ 9,9 bilhões – valor 32% maior do que o movimentado no ano de 2004, que foi de R$ 7,5 bi, e correspondente a 3,43% do varejo total no país - dados estimados a partir do índice-base do IBGE[CAMARA, 2006].

Conforme Almeida [Almeida,1999] este mercado não para de crescer e enquanto o rádio demorou cerca de 35 anos para atingir 60 milhões de pessoas e a televisão 15 anos, a Internet levou apenas 6 anos para atingir 100 milhões de pessoas em praticamente todo o planeta. Nos EUA, mais de 90% das grandes companhias já conduzem algum tipo de negócio através da Internet, enquanto mais de 400 mil empresas, do mundo inteiro, já têm sites comerciais na Web.

Dentro do contexto informacional da Internet, verificamos que o comércio eletrônico é definido como sendo a compra e a venda de informações, produtos e serviços através de redes de computadores [Mendes, 1999].

Segundo Siegel [SIEGEL, 2000], o comércio eletrônico - e-commerce - significa colocar os catálogos nos sites e receber os pedidos on-line. Isto envolve uma transação e um carrinho de compras virtual; os formulários de serviços do e-commerce fornecem aos clientes acesso a relatórios, dados, conselhos e outras informações, mediante uma assinatura paga ou gratuita, em troca de visitas, a fim de expor os anúncios.

A grande maioria dos sites de cunho comercial ainda se encontra na fase de promoção da empresa ou do negócio, descrevendo sua atividade e, às vezes, apresentando produtos com respectivos preços, mas sem oferecer possibilidade de concluir a transação on-line.

2.2. E-BUSINESS

O negócio eletrônico - e-business - é uma abordagem mais focalizada e mais abrangente do que o e-commerce, sendo sua meta fornecer a públicos específicos uma experiência completa e personalizada, ou seja, fornecer aos clientes o poder de conseguir “o que querem, quando querem e da forma que querem”. Desta forma, uma empresa em uma plataforma e-business responde mais apropriadamente às novas demandas do cliente [Seigel, 2000].

Combinando recursos computacionais com o alcance global da Internet, o e-business tornou-se uma maneira dinâmica e interativa de fazer negócios, oferecendo inúmeras oportunidades, explorando relações informatizadas, tempo de resposta eletrônico, operações virtuais e automação, para obter vantagens competitivas, visando, desta forma, a um melhor atendimento aos clientes, a uma redução do ciclo do produto e à venda destes produtos.

O e-business começa a ser praticado a partir do momento em que uma empresa cria sua home page na Internet, expondo informações a seu respeito (nome, localização, ramo de negócio, telefones para contato, produtos e/ou serviços). Contendo apenas informações acerca da empresa e telefones e/ou endereços para contato, trata-se apenas de mais um instrumento de marketing. Mas a partir do momento em que na página da empresa exista um link para que o visitante possa enviar uma mensagem eletrônica para esta empresa, contendo sugestões ou dúvidas sobre determinado produto ou serviço, detecta-se a prática do e-business [Seigel, 2000]).

O e-business inclui o e-commerce, no entanto, este não favorece o diálogo entre funcionário e cliente on-line, limitando o contato entre a equipe Web e clientes. Este diálogo proporcionaria um relacionamento mais aprofundado e encorajaria a fidelidade do cliente. No e-business, onde o negócio é conduzido pelo cliente, há uma interação maior entre os funcionários e os clientes [Seigel, 2006].

Endossando essa idéia, segundo Tapscott [ALEXANDRINI,2000]

> **A nova economia está se transformando em "Economia Digital" ou "Economia do Conhecimento", ou seja, a economia da era da inteligência em rede, onde o fluxo de informação não é mais físico como na velha economia. Dinheiro, cheques, faturas, notas de embarque, relatórios, reuniões face a face, mapas, fotografias e outros serão substituídos por informações digitais, ou melhor, bits de computador.**

Com isto, os vendedores podem atingir um número maior de clientes, especialmente aqueles que jamais seriam atingidos sem o uso de tal tecnologia. Isso possibilita oferecer produtos com menor custo. Outro motivo para a redução dos custos é porque a Internet proporciona o aparecimento de compradores com melhores informações sobre características dos produtos.

Fazer negócios ficou mais simples e com menos custos. Isso possibilitou que as empresas pensem globalmente, façam negócios sem limitação dos aspectos geográficos e até mesmo culturais.

As empresas precisam reestruturar suas estratégias de marketing inventando novas maneiras de criar, comunicar e transmitir valor a seus mercados-alvo, [KOTLER, 1999].

Neste sentido é necessário criar um ambiente de negócios caracterizado por rapidez e mudanças radicais. Isso exige um modelo de negócios inovador que ofereça aos clientes novidades e uma competitiva e sustentável proposição de valor voltada para o cliente. Estabelecendo uma agenda para digitalização de suas empresas, executivos de tecnologia necessitam reconhecer que suas empresas podem criar um modelo de E-Business viável, somente para atender os fundamentos da agilidade e flexibilidade [INTEL, 2001].

As soluções de E-Business se preocupam com toda a infra-estrutura tecnológica existente em uma organização, entre eles a automação de força de vendas.

2.2.1. Arquitetura de Ebusiness

A nova geração de E-Business marcará as questões de integração e extensão de aplicações empresariais. O foco deve estar em velocidade e flexibilidade. Para isso as empresas precisam de tecnologias de E-Business exteriores, providos por “plug in play“ e capacidades tecnológicas que permitam adaptações nos modelos de negócios. Isso exige mais máquinas e máquinas (M2M) que permitam checar questões relacionadas, estoques de produtos, logística de distribuição, créditos e ficha financeira dos clientes, otimizando tempo de entrega e minimizando custos [INTEL, 2001].

No aspecto técnico as integrações entre os dados locais dos negócios dos clientes com os múltiplos e remotos serviços dos vendedores formam a base para um E-Business centrado nos clientes. Essa arquitetura deve também oferecer mecanismos de entrega dos dados que descrevem os mais variados níveis de complexidade. Outro aspecto que deve ser tratado são as perdas de conexões que ocorrem durante um processo de transmissão

de dados. Essa arquitetura suporta a tecnologia “M-Commerce” onde os usuários podem trabalhar “of line” e sincronizar em uma próxima conexão.

Enquanto as arquiteturas de serviços serão específicas para os relacionamentos entre vendedores e clientes, elas deverão ser suportadas por tecnologias chaves tais como XML, COM, Java, CORBA [INTEL, 2001].

Dependendo de como a empresa lida com as questões de aplicabilidade de tecnologias, poderá obter sucesso ou fracasso nos resultados esperados. Para se ter sucesso, na prática, significa estar um passo a frente das necessidades dos clientes oferecendo possibilidades de se realizar negócios com menor custo e com maior agilidade. A arquitetura para E-Business deverá estar centrada no cliente, onde este terá papel importante guiando os negócios das empresas definindo as maiores preposições de valores dos clientes.

2.2.2. E-Commerce

Estamos vivenciando uma nova forma de se fazer negócios, caracterizada por ser a grande responsável pela democratização das negociações comerciais.

Segundo [Kotler ,1999], as transações comerciais em um futuro próximo serão realizadas por intermediação de atacadistas e varejistas, sofrendo uma redução considerável devido ao comercio eletrônico. Virtualmente, todos os produtos estão disponíveis sem que seja necessário ir à loja. O comércio eletrônico é definido por Ribeiro [ALEXANDRINE, 2001] como “atividade mercantil, que em última análise, vai fazer a conexão eletrônica entre a empresa e o cliente.”

Patrícia [SEYBOLD, 2002] define comércio eletrônico como “o ato de fazer negócios eletronicamente”. Ela complementa que “o comércio eletrônico engloba todo o processo de negócios (propaganda, marketing, vendas, pedidos, manufatura, distribuição, serviço ao cliente, suporte pós-venda, reposição de estoques, etc.) envolvendo várias tecnologias emergentes além da Internet.”

Em expansão no mundo, o comercio eletrônico constitui-se na forma mais avançada de venda direta ao consumidor, tendo a Internet como o principal meio pelo qual se adquirem bens e serviços à distância, por transmissão e recepção de dados [Melo, 2002].

O surgimento dos Portais de negócios, por exemplo, possibilitou com que pequenas empresas pudessem realizar suas compras com agilidade e custo reduzido..

Apesar de ser proporcionado pelo desenvolvimento tecnológico principalmente com as melhorias nas telecomunicações e com a difusão da Internet, o ciclo de vida do comércio eletrônico na perspectiva do comprador e do vendedor estão baseadas no princípio mais elementar do comércio “as trocas” [ALEXANDRINE, 2001].

Para Melo [Melo, 2002] “o incremento do comércio eletrônico mundial é uma realidade e as oportunidades deste comércio virtual na Internet são ilimitadas, tendo em vista as soluções criativas e dinâmicas da alta tecnologia.” A tecnologia E-COMMERCE é uma opção para automação de força de vendas, principalmente para os casos onde as transações comerciais são realizadas diretamente entre as empresas.

2.3. WIRELESS

Wireless (sem fio) ou Wi-fi (Wireless Fidelity) é o termo usado para receptores de rádios. O termo começou a ser usado no Reino Unido, logo depois que uma rádio começou a transmitir para outros sinais [WIKIPEDIA, 2006a].

Um protocolo de comunicação sem fios é desenhado com o objetivo de criar redes wireless de alta velocidade e que não fazem mais do que transferir dados por ondas de rádio em freqüências não licenciadas.

E é precisamente pelo fato de serem freqüências abertas, que não necessitam de qualquer tipo de licença ou autorização do regulador das comunicações para operar, ao contrário das demais áreas de negócio, o que as torna tão atrativas.

No entanto, para uso comercial é necessário, no Brasil, licença da Agência Nacional de Telecomunicações -Anatel.

Este tipo de tecnologia pode ser utilizada para acesso à Internet através de dispositivos móveis.

O funcionamento do 'Wi-Fi' é simples. Para se ter acesso à Internet através de uma rede Wi-Fi (também conhecida como Wlan) deve-se estar no raio de ação de um ponto de acesso (normalmente conhecido por hotspot) ou local público onde opere uma rede sem fios e usar um dispositivo móvel, como um computador portátil, um Table PC ou um assistente pessoal digital (PDA) com capacidades de comunicação Wireless [WIKIPEDIA, 2006a].

Um Hotspot 'Wi-Fi' é criado para estabelecer um ponto de acesso para uma conexão de Internet. O ponto de acesso transmite um sinal sem fio numa pequena distância – cerca de 100 metros. Quando um periférico que permite 'Wi-Fi', como um Pocket PC, encontrar um hotspot, o periférico pode na mesma hora conectar na rede sem fio. Muitos hotspots estão localizados em lugares que são confortavelmente acessíveis ao público, como aeroportos, cafés, hotéis e livrarias. Muitas casas e escritórios também têm redes 'Wi-Fi'. Enquanto alguns hotspots são gratuítos, a maioria das redes públicas é suportada por Provedores de Serviços de Internet (Internet Service Provider - ISPs) que cobram uma taxa dos usuários para conectar na Internet.

Para os portáteis mais recentes, a Intel, maior fabricante mundial de microprocessadores, já fornece um pacote de Rede Wireless Centrino. Para os que não venham equipadas com este pacote, as soluções é recorrer às mais diversas placas e cartões especialmente desenvolvidos para o efeito.

Desenvolvedores de software e hardware estão criando computadores menores que formam uma Internet sem fio ad-hoc, com protocolos como Wifi. A IEEE 802.11 padronizou o uso wireless de conexões locais [WIKIPEDIA, 2006a]

2.4. M-COMMERCE

Müller-Veerse [MOSKORS, 2002] define M-Commerce como sendo "qualquer transação com um valor monetário que é conduzido via rede de telecomunicações móveis."

Para [Moskors 2002] "M-Commerce é a sigla para MobilECommerce. Numa tradução livre, poderia se chamar de Comércio Móvel." O correto seria E-M-Commerce explicitando o conceito de "Comércio Eletrônico Móvel."

Nota-se que M-Commerce é uma extensão da tecnologia E-Commerce. Porém Siau, Lim, Shen [MOSKORS, 2002] explicam que existem grandes diferenças entre as duas tecnologias, a começar pelo fato de o M-Commerce utilizar redes de telecomunicações sem fios. O M-Commerce é caracterizado como sendo a tecnologia que oferece ao usuário alto grau de acesso às informações onde quer que esteja; acessa apenas as informações desejadas; conduz transações comerciais em quanto se locomove de um lugar para outro e a possibilita de entrega simultânea dos dados a todos os usuários em uma área geográfica específica. Fica claro que essas características diferem daquelas predominantes na Internet com fios.

Para [WMW ,2005] M-Commerce é a criação de valor por intermédio de dispositivos móveis e com rádio freqüência, disponibilizando comunicação, acesso às informações e as transações de negócios. M-Commerce usa essa capacidade de redes móveis para conectar computadores PDAs, telefones, carros e aplicações domésticas.

No aspecto tecnológico o uso da tecnologia M-Commerce oferece aos negociadores uma ferramenta computacional que possibilita acesso às informações em tempo de negociação. Os PDAs, coloquialmente conhecidos como PalmTops, são as ferramentas

ideais para aplicação de softwares que caracterizem o MCommerce, permitindo assim implementar os conceitos amplamente discutidos na gestão comercial.

As soluções de “M-Commerce” objetivam prover de informações aos homens de negócios, que poderão contar com tais informações estratégicas para a tomada de decisões junto ao cliente. Isso facilitará a negociação entre representantes comerciais e os clientes.

As soluções de M-Commerce oferecem a possibilidade de se realizar negociações utilizando-se da tecnologia de Internet e um computador de mão. O que realmente diferencia e caracteriza essa tecnologia é o fato de que a informação necessária para o agente de negócios está sempre a sua disposição e com facilidade de acesso. Ter a informação no momento da negociação muitas vezes faz a diferença.

Dessa forma uma solução de M-COMMERCE para automação da força de vendas é parte de uma solução de E-BUSINESS, sendo relevante o estudo desse tema nesta pesquisa.

2.5. COMUNICAÇÕES CELULARES

2.5.1. Evolução das Comunicações Celulares

O desenvolvimento das comunicações celulares ocorreu em etapas ou gerações. A partir de sua primeira geração o serviço celular passou a funcionar através da divisão de uma cidade ou região em pequenas áreas geográficas denominadas células, sendo cada uma delas servida pelo seu próprio conjunto de rádios transmissores e receptores de baixa potência. Quando a chamada de um celular alcança uma torre de transmissão e recepção, a mesma é transferida para o sistema de telefonia fixa regular. Cada célula possui diversos canais com o objetivo de prover serviços para muitos usuários simultaneamente. Na medida em que um usuário se movimenta na cidade, o sinal do seu telefone celular passa automaticamente de uma célula para outra, sem sofrer interrupção.[NASCIMENTO, 2000]

Os sistemas da primeira geração eram bastante parecidos entre si, sendo que as principais diferenças concentravam-se no uso do espectro de freqüência e no espaçamento entre canais. O AMPS (Advanced Mobile Phone System), por exemplo, opera na faixa de 869-894 MHz para recepção e 824-849 MHz para transmissão; o NMT-450 (Nordic Mobile Telephone) opera na faixa de 463-468 MHz para recepção e 453-458 MHz para transmissão enquanto que o NMT-900 utiliza a faixa de 935-960 MHz para recepção e 890-915 MHz para transmissão, etc. Com relação ao espaçamento entre os canais pode-se citar, por exemplo, o AMPS que adota 30 kHz, o TACS (por extenso) e vários outros que adotam 25 kHz, etc. [WIRELESSBRASIL, 2006]

Essa primeira geração de sistemas celulares caracterizava-se basicamente por ser analógica, utilizando modulação em freqüência para voz e modulação digital FSK (Frequency Shift Keying) para sinalização. O acesso à canalização é obtido através do FDMA (Frequency Division Multiple Access) [WIRELESSBRASIL, 2006].

A segunda geração toma como base o desenvolvimento de sistemas digitais que em princípio, além da maior capacidade, oferecia as seguintes vantagens sobre os analógicos: técnicas de codificação digital de voz mais poderosas, maior eficiência espectral, melhor qualidade de voz, trabalham com bastante facilidade a comunicação de dados e facilitam significativamente a criptografia da informação transmitida. [WIRELESSBRASIL, 2006]

Como resultado desse esforço, surgiram os sistemas GSM (Groupe Speciale Mobile/Global System for Mobile Communications) na Europa, o TDMA (Time Division Multiple Access), o CDMA (Code Division Multiple Access) nos EUA e o PDC (Japanese Personal Digital Cellular) no Japão [WIRELESSBRASIL,2006].

O TDMA opera dividindo o tempo de um canal, que opera em uma determinada freqüência, em certo número de partes e designando cada uma das diversas conversações telefônicas para cada uma dessas partes.

O sistema CDMA utiliza a técnica de espalhamento espectral e foi originalmente utilizado pelos militares para espalhar o sinal em uma faixa de espectro bastante larga, tornando as transmissões difíceis de interceptar ou mesmo interferir.

O GSM foi adotado como padrão Europeu em meados dos anos 80 e introduzido comercialmente em 1992, operando na faixa de freqüência 935-960 MHz para recepção e 890-915 MHz para transmissão. O GSM possui uma arquitetura aberta, o que permite a combinação de equipamentos de diferentes fabricantes, possibilitando assim a manutenção de preços baixos. O GSM é hoje, o padrão mais popular implementado mundialmente, razão pela qual a escolha desta tecnologia é justificada neste projeto [WIRELESSBRASIL, 2006].

Em 1989, na Ingalterra, surge o conceito PCN (Personal Communications Network), que tomava como base um sistema de rádio que fornecendo serviços bidirecionais de telecomunicações de alta qualidade, para ambientes fixos e móveis, a um custo acessível. A arquitetura do sistema seria suportada por uma ampla estrutura microcelular para possibilitar o uso de terminais de baixa potência e, conseqüentemente, leves para serem transportados no bolso (pocket-size). A faixa de freqüência mais adequada estaria entre 1,7 e 2,3 GHz, por estar menos congestionada que a faixa do celular convencional, em torno dos 900 MHz, e a atenuação adicional da nova faixa seria compensada pela menor dimensão das células [WIRELESSBRASIL,2006].

O termo PERSONAL ou PESSOAIS é visto como ponto-chave em termos mercadológicos porque captura a imaginação e inspira liberdade, individualidade e algo feito sob medida. As operadoras vêem nessa solução uma forma de melhorar os serviços já

oferecidos onde se incluem atualmente os celulares, os "pagers" e a própria rede fixa de telefonia convencional.[WIRELESSBRASIL, 2006]

Na Europa, as primeiras aplicações de PCS surgiram no final de 1993 com o sistema DCS-1800, uma variante do GSM (Global System for Mobile Communications, ou Sistema Global para Comunicações Móveis) operando com potências menores e em uma faixa de freqüência mais alta..

Progressos significativos já foram obtidos no que se denomina a terceira geração que prevê a criação de um sistema móvel universal um exemplo disso é a reserva de 230 MHz de espectro, com a aprovação de 127 países, na "World Administrative Radio Conference" (WARC) em 1992 [WIRELESSBRASIL, 2006].

A topologia provável desse novo sistema será baseada em uma forma de arquitetura mista de células; células de tamanho variável serão implementadas com dimensionamento adequado para áreas geográficas específicas e em função das diferentes demandas de tráfego. Células diminutas, ou seja, picocélulas, instaladas em interiores, serão versões melhoradas das atuais tecnologias "cordless", com "handsets", isto é, aparelhos de assinante, bastante pequenos e leves; células maiores, ou seja, microcélulas e macrocélulas, poderão operar segundo características evoluídas a partir do GSM. "Handsets" diferentes precisarão reconhecer e operar indistintamente em pico, micro e macrocélulas. Ou seja, o objetivo é criar uma plataforma de rede SEM FIO, oferecendo aos usuários a possibilidade de acesso, através de ondas de rádio, como extensão do sistema telefônico do escritório quando se encontram no trabalho ou como telefone móvel convencional quando se encontram ausentes ou ainda como telefone principal de suas residências quando estão em casa [WIRELESSBRASIL, 2006].

A evolução em direção aos serviços de telecomunicações móveis universais, UMTS, muito provavelmente, deverá ter como base a estrutura do GSM. [WIRELESSBRASIL, 2006].

2.5.2. Rede GMS

As redes GSM (Global System for Mobile Communications) surgiram da tentativa do European Telecommunication Standards Institute (ETSI) de padronizar os sistemas celulares privados que os países europeus vinham desenvolvendo na década de 1980. Estes sistemas eram incompatíveis e suportavam apenas “roaming” nacional, limitando fortemente o mercado de celulares na Europa. O GSM foi desenvolvido para suportar boa qualidade de voz, baixo custo, compatibilidade com ISDN e “roaming” por toda a Europa. [WIRELESSBRASIL, 2006]

No entanto, o GSM cresceu muito mais do que o esperado e hoje é usado por mais de 200 países em todo o mundo.

Para cumprir seus objetivos, o padrão GSM foi projetado para ser um sistema digital baseado em comutação de circuitos, fazendo com que cada chamada de um usuário a outro ocupe exclusivamente um canal. Por ser um sistema digital, técnicas de compressão e processamento de sinais podem ser usadas para reduzir custos e aumentar a eficiência do sistema.

2.5.2.1. Arquitetura GSM

A arquitetura de referência de um sistema GSM é apresentada na figura a seguir:

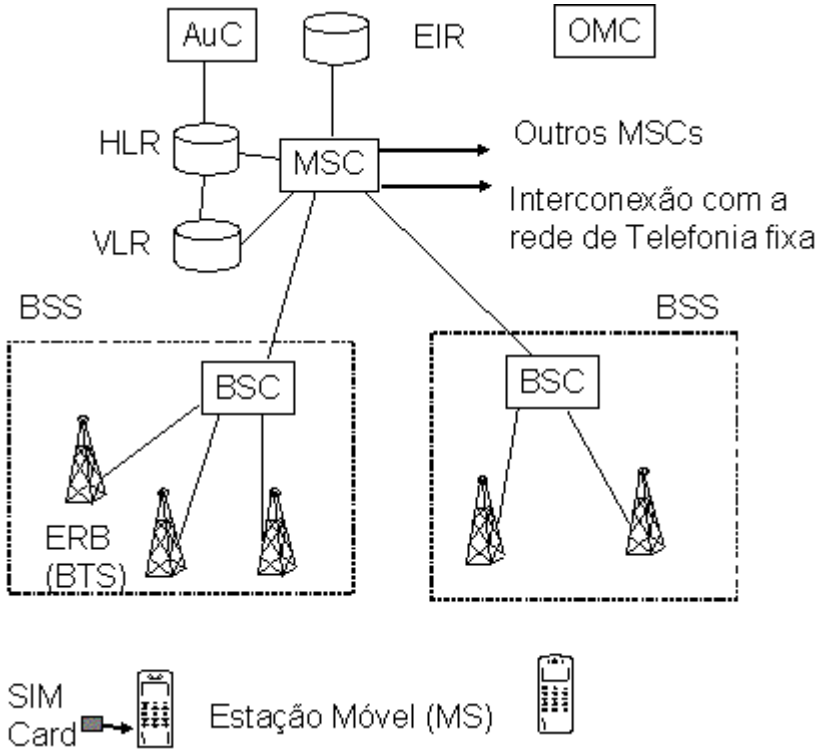

Figura 2.1 - Arquitetura GSM [TELECO, 2005]

A figura 2.1 mostra os componetes da arquitetura GSM.

A Estação Móvel (MS) é o terminal utilizado pelo assinante quando carregado com um cartão inteligente conhecido como SIM Card ou Módulo de Identidade do Assinante (Subscriber Identity Module). Sem o SIM Card a Estação Móvel não está associada a um usuário e não pode fazer nem receber chamadas.

A Estação Base (BSS) é o sistema encarregado da comunicação com as estações móveis em uma determinada área. É formado por várias Base Transceiver Station (BTS) ou ERBs, que constituem uma célula, e um Base Station Controller (BSC), que controla estas BTSs.

A Central de Comutação e Controle (CCC ou em inglês MSC) é a central responsável pelas funções de comutação e sinalização para as estações móveis localizadas em uma área geográfica designada como a área do MSC.

A base de dados que contém informações sobre os assinantes de um sistema celular, está representada na figura 2.1 pelo (HLR – Home Location Register)

O Registro de Assinantes Visitantes (VLR - Visitor Location Register) é a base de dados que contém a informação sobre os assinantes em visita (roaming) a um sistema celular.

O Centro de Autenticação (AUC – Authentication Center) é responsável pela autenticação dos assinantes no uso do sistema e o Registro de Identidade do Equipamento (EIR – Equipment Identity) é a base de dados que armazena os IMEIs dos terminais móveis de um sistema GSM.

O Centro de Operação e Manutenção (OMC – Operational and Maintenence) é a entidade funcional através da qual a operadora monitora e controla o sistema.

2.5.2.2. Teleservices

São serviços de comunicação entre dois assinantes como telefonia, serviço de mensagens curtas (SMS) e FAX.

As aplicações iniciais de SMS visavam eliminar os pagers alfanuméricos, permitindo serviços em que as mensagens poderiam ser enviadas nos dois sentidos, primeiramente para o correio de voz. As tecnologias de redes evoluíram e uma variedade de serviços foram lançados, incluindo o e-mail, o fax e a integração com paging, operações bancárias, serviços de informação, tais como indicadores econômicos, e integração com aplicações baseadas na Internet.

Nos dias atuais, além de poder mandar um recado para o celular de outra pessoa, o usuário também está submetido a vários diferentes serviços disponibilizados pelas operadoras. O que as prestadoras estão trazendo agora é uma interatividade maior, algo além de, unicamente, o texto.

Uma mensagem de texto SMS é uma string de tamanho muito pequeno, de 100 a 256 caracteres. Para aplicações na telefonia celular o comprimento mais comum é 150 caracteres que trafegam nos canais de controle.

Os serviços de mensagens estão a tornar-se um veículo de informação e úteis nas transações de serviços. Neste cenário, os celulares estão, rapidamente, se transformando em dispositivos mais avançados tornando a telefonia móvel uma nova era aplicacional.

Multimedia Messaging Service (MMS) é o próximo passo na evolução dos sistemas de mensagens, tornando-se as principais aplicações nas redes móveis com a evolução da largura de banda para o GPRS, EDGE e UMTS.

Este projeto optou em utilizar a tecnologia SMS, pelo fato que as mensagens trocadas durante o processo descrito no projeto, são pequenas, a tecnologia é suportada pela maioria dos aparelhos GSM além de ser um serviço relativamente barato.

A opção por GSM é justificada pela figura abaixo onde é demonstrada graficamente a participação da tecnologia GSM no Brasil.

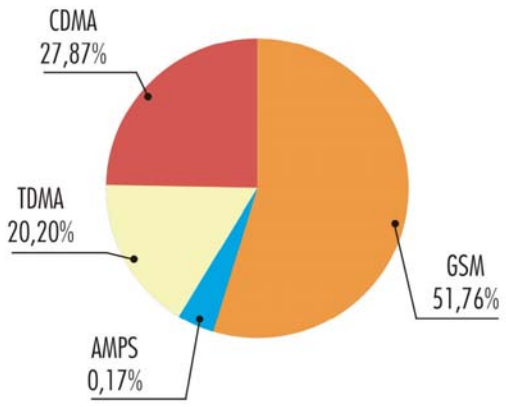

Figura 2.2 – Participação da tecnologia GSM no mercado de telefonia celular no Brasil [ANATEL, 2005].

2.5.3. GPRS

Com a popularização do uso da Internet e de outros serviços de dados em meados da década de 1990, previu-se que as redes GSM não seriam capazes de suportar esta demanda.

Em 1997, entretanto, o ETSI publicou o modo de funcionamento do GPRS (General PacketRadio Service) na especificação da FASE 2+ do GSM. As redes GPRS foram desenvolvidas para suportar os serviços de dados, pois foram criadas baseadas em transmissão por comutação de pacotes, diferentemente das GSM. A comutação de pacotes utiliza mais eficientemente a fonte de rádio para tráfego em rajadas, como é característica da maioria dos serviços de dados. [Bettstetter, 1999]

2.5.4. GSM x GPRS

Para que as operadoras possam utilizar seus serviços GSM e os serviços de dados GPRS a partir de uma única base dinâmica e flexível, os dois sistemas compartilham várias características entre si, como bandas de freqüência, estrutura de frames e técnicas de modulação. A interface aérea GPRS é a mesma interface utilizada nas redes GSM, isto é, uma combinação de FDMA (Frequency Division Multiple Access) e TDMA (Time Division Multiple Access) [Bettstetter, 1999].

As taxas máximas de transmissão de dados estão resumidas na Tabela 2.3 Em razão do GSM ser adaptado para transmissão de voz, as taxas de transmissão de dados alcançadas são relativamente baixas. O padrão GSM é um sistema de comutação de circuitos, e por isso tipicamente reserva um canal inteiro de tráfego por usuário, mas este pode utilizar apenas 1 slot de tempo por frame TDMA para tráfego.

Tabela 2.1– Taxas máximas de transmissão. [WATKINS, 2000]

Sistema	Taxa máxima de Dados
GSM	14.4 kbps
GPRS	171.2 kbps
UMTS	144 kbps – veículos móveis 384 kbps – usuários móveis 2 Mbps – ambiente interno

Em contrapartida, o GPRS é adaptado para tráfego de dados, pois é implementado sobre um sistema de comutação de pacotes. Sendo assim, os canais de tráfego são reservados para o usuário apenas durante o tempo de sua transmissão. Por esta razão, o GPRS é capaz de atribuir todos os 8 slots de tempo de um frame TDMA para um único usuário, dependendo da carga no sistema. Desta forma, seria possível alcançar até 115.2 kbps (14.4 x 8) [WATIKINS, 2000] uma taxa ainda baixa em comparação com redes fixas. Porém, GPRS suporta quatro esquemas de codificação diferentes, escolhidos de acordo com a condição do canal.

Como resultado das diferentes taxas de transmissão alcançadas, cada um dos sistemas é capaz de suportar diferentes serviços. Os serviços suportados pelo GSM podem ser divididos em 3 categorias: tele serviços (telefonia, SMS, correio de voz), transporte (transporte síncrono e assíncrono de dados de usuários) e suplementares (chamada em espera, redirecionamento de chamada).

O sistema GPRS tem como característica o tráfego de dados em rajadas, podendo assim suportar a maioria dos serviços encontrados na Internet, além de SMS, transmissão de dados point-to-point (PTP) e point-to-multipoint (PTM) e Qualidade de Serviço (QoS).

Infelizmente as taxas de transmissão oferecidas pelo GPRS ainda não são suficientemente altas para serviços multimídia.

Atualmente a cobrança pelo uso de GPRS é feita com base na quantidade de dados transmitidos enquanto no GSM é feita por tempo de conexão.

A rede GPRS pode ser considerada como um revestimento à rede GSM, acrescentando tráfego orientado a pacotes mediante leves modificações na arquitetura, logo é comumente dito que a arquitetura GPRS pode ser analisada como sendo “GSM + dados”.

O projeto foi estruturado para utilizar tecnologia GSM e mensagens SMS.

2.6. WIG APPLICATION CREATOR (WAC)

O WIG Application Creator, uma caixa de ferramentas para a simulação de serviços em ambientes móveis baseados em rede GSM, SmartTrust Delivery Platform e SmartTrust WIB (Wireless Internet Browser) criado pela empresa SmartTrust [SMARTTRUST, 2006].

O WIG é uma ferramenta autônoma fornecida em duas versões, a Standard e a Professional que possui um editor de texto próprio para a programação de aplicações WML(Wireless Markup Language, além do WIB Phone, um simulador de aplicações em ambiente WIB, o navegador wireless para telefone celular da SmartTrust.

A ferramenta também busca e simula aplicações WML diretamente de um endereço URL no web Server e transfere e valida aplicações em um terminal real, além de suportar aplicações WIG WML incluindo simulações de plug-ins do SmartTrust WIB.

Do lado da aplicação, a caixa de ferramentas tem a mesma interface HTTP para o Wireless Internet Gateway (WIG) e as sintaxes dos desks WML podem ser manualmente ou automaticamente auditadas por uma ferramenta e qualquer sintaxe defeituosa é destacada com um indicador de erro, imediatamente. Esta ferramenta suporta o desenvolvimento de aplicações WML com caracteres Unicode (UTF-8).

"O SmartTrust WIG Application Creator é uma aplicação autônoma de desenvolvimento, que não necessita de um servidor WIB e SIM Cards externos para os testes de validação das aplicações desenvolvidas", ressalta Alexander Dannias, diretor de vendas para a América Latina e Caribe da SmartTrust.

A ferramenta está disponível na WEB em sua versão Standard para que desenvolvedores e operadoras possam baixá-la sem custo em http://www.smarttrust.com/wac/, o que ratifica o conceito de linguagem aberta e business model não proprietário das soluções oferecidas pela SmartTrust. "Esta iniciativa tem contribuído para facilitar o acesso a grande variedade de aplicações disponíveis para o Wireless Internet Browser, suportado pela Plataforma OTA da SmartTrust", informa Dannias[SMARTTRUST, 2006].

2.7. WML

A linguagem WML (“Wireless Markup Language”) [SMARTTRUST, 2006] é uma linguagem de construtores que estabelecem regras para a organização, navegação, visualização e interação com conteúdos de informação, como textos, “hyperlinks”, formulários e imagens.

É uma linguagem similar à HTML [SMARTTRUST, 2006], adaptada às condições dos terminais do ambiente sem fio visto anteriormente.

As quatro principais áreas funcionais da WML são:

- Apresentação de conteúdos: WML inclui suporte à apresentação de textos, imagens e tabelas, possuindo comandos de formatação para texto (posição, tamanho e forma dos caracteres), tabelas (posição e tamanho) e imagens (posição e tamanho).
- Organização lógica em unidades de conteúdo: Toda informação em WML é organizada em coleções (“decks”) de unidades de conteúdo

(“cards”). Cada unidade possui o conteúdo a ser apresentado ao usuário, especificando uma ou mais unidades de interação, como um menu de escolha, um campo para a entrada de texto, um “link” ou outros.

- Navegação através de conteúdos referenciados: Através de comandos, a WML permite a troca de unidades de conteúdo relacionadas com a unidade atual. Tal navegação pode ser resultado da decisão do usuário ou de um evento ocorrido (expiração de tempo, por exemplo).

- Parametrização de textos e gerenciamento de estados: WML permite o uso de variáveis no lugar de textos, permitindo sua substituição em tempo de execução e tornando mais eficiente o uso dos recursos da rede.

Isto é, a rotina de uso esperada é o usuário navegando através de unidades de conteúdo, recebendo as informações contidas nelas e fornecendo as informações solicitadas (dados ou ações). WML assume a mesma referência de arquitetura do HTML, identificando conteúdos usando URLs e buscando através de protocolos com semântica similar à do HTTP, como WSP [SMARTTRUST, 2006]. Em WML, URLs são usadas nas seguintes situações:

- Quando especificando navegação de conteúdos (“hyperlink”)
- Quando especificando recursos externos (imagens ou “scripts”).

O mais importante sobre o WML, desde que é uma língua XML-definida, é "Bem Formada" (Well-Formed). Isto significa que, diferente do HTML, todos as tag´s devem ser fechadas e não podem ser incorretamente alinhadas, e todos os valores dos atributos devem ser incluídos nas marcas da citação da tag.

WML também é validado por um DTD, que significa que todo o índice de WML está verificado em uma “Definição do Tipo de Documento” de WML antes que seja

mostrado. Assim os tag´s devem ter exatamente os atributos especificados no DTD e não podem ser fechados anterior uma outra tag a menos que permitido pelo DTD.

2.8. JAVA (LINGUAGEM DE PROGRAMAÇÃO)

2.8.1. Principais Características da Linguagem Java

A linguagem Java foi projetada tendo em vista os seguintes objetivos:

- Orientação a objeto - Baseado no modelo de Smalltalk e Simula67;
- Portabilidade - Independência de plataforma - "write once run anywhere";
- Recursos de Rede - Possui extensa biblioteca de rotinas que facilitam a cooperação com protocolos TCP/IP, como HTTP e FTP;
- Segurança - Pode executar programas via rede com restrições de execução;
- Bytecode interpretado, ao invés de compilado.

Além disso, podem-se destacar outras vantagens apresentadas pela linguagem:

- Sintaxe similar a Linguagem C/C++.
- Facilidades de Internacionalização - Suporta nativamente caracteres Unicode;
- Simplicidade na especificação, tanto da linguagem como do "ambiente" de execução (JVM);
- É distribuída com um vasto conjunto de bibliotecas (ou APIs);
- Possui facilidades para criação de programas distribuídos e multitarefa (múltiplas linhas de execução num mesmo programa);
- Desalocação de memória automática por processo de coletor de lixo;

- Carga Dinâmica de Código - Programas em Java são formados por uma coleção de classes armazenadas independentemente e que podem ser carregadas no momento de utilização [WIKIPEDIA, 2006b].

2.8.2. Máquina Virtual Java

Programas Java não são traduzidos para a linguagem de máquina como outras linguagens estaticamente compiladas e sim para uma representação intermediária, chamada de bytecodes.

Os bytecodes são interpretados pela máquina virtual Java (JVM - Java Virtual Machine). Muitas pessoas acreditam que por causa desse processo, o código interpretado Java tem baixo desempenho. Durante muito tempo esta foi uma afirmação verdadeira. Porém novos avanços tem tornado o compilador dinâmico (a JVM), em muitos casos, mais eficiente que o compilador estático.

Java hoje já possuiu uma performace próxima do C++. Isto é possível graças a otimizações como a compilação especulativa, que aproveita o tempo ocioso do processador para pré-compilar bytecode para código nativo. Outros mecanismos ainda mais elaborados como o HotSpot da Sun, que guarda informações disponíveis somente em tempo de execução (ex.: número de usuários, processamento usado, memória disponível), para otimizar o funcionamento da JVM, possibilitando que a JVM vá "aprendendo" e melhorando seu desempenho. Isto é uma realidade tão presente que hoje é fácil encontrar programas corporativos e de missão crítica usando tecnologia Java

Os bytecodes produzidos pelos compiladores Java podem ser usados num processo de engenharia reversa para a recuperação do programa-fonte original. Esta é uma característica que atinge em menor grau todas as linguagens compiladas. No entanto já

existem hoje tecnologias que "embaralham" e até mesmo criptografam os bytecodes praticamente impedindo a engenharia reversa [WIKIPEDIA, 2006b].

2.9. SERVIDOR APACHE TOMCAT

O servidor Apache Tomcat, que foi desenvolvido pela Fundação Apache é um servidor de Aplicações que também é utilizado como servidor web.

Como servidor de aplicação, o Tomcat deve ser integrado a um servidor web como Apache, IIS, Websphere, dentre outros[WIKIPEDIA, 2006c].

2.9.1. O que é o Tomcat

O Tomcat é um servidor de aplicações desenvolvido pela Fundação Apache utilizando tecnologia Java. O Tomcat foi desenvolvido para substituir o Servidor Jserv, que estava apoiado em tecnologias ultrapassadas.

Sua principal característica técnica é estar centrado na linguagem de programação Java, mais especificamente nas tecnologias de Servlets e de JSP (Java Server Pages).

Como o Tomcat foi escrito em Java ele necessita que a versão J2SE (Java 2 Standard Edition) esteja instalada no mesmo computador onde ele está executando para que possa compilar programas escritos em Java.

Para se desenvolver uma típica aplicação web que será executada pelo Tomcat é necessário o domínio das seguintes linguagens:

- Java – os algoritimos da aplicação devem estar escritos em Java;
- HTML – para criar páginas que serão visualizadas pelo browser do usuário;
- XML – para gerar os dados de configuração que podem ser usados tanto pelo Tomcat como pela aplicação.

Outra característica importante do Tomcat é que ele tem a habilidade de converter automaticamente qualquer programa JSP em um servlet equivalente, isto é, ele é capaz de criar código fonte Java a partir de um documento HTML.

Do ponto de vista prático, o Tomcat pode ser usado isoladamente como um servidor web ou juntamente com outro servidor (como Apache). Na Segunda opção o Apache atende a requisições de páginas estáticas enquanto que o Tomcat atende a requisições de páginas dinâmicas, como ilustra a Figura 2.3 .

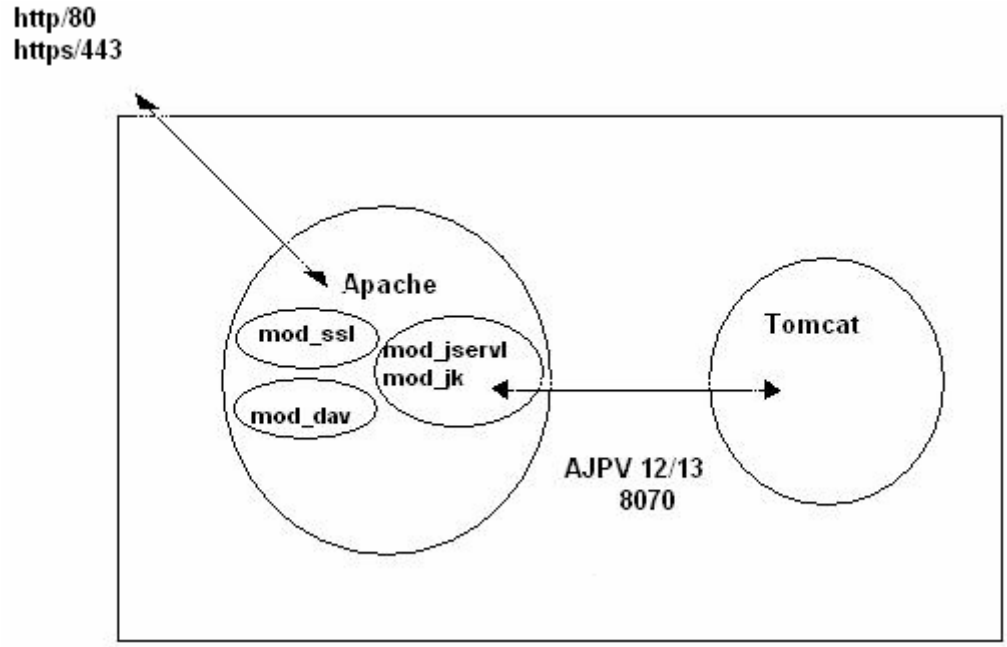

Figura 2.3 – Servidor Apache Tomcat

Uma outra utilização do Tomcat é como parte da versão J2EE (Java 2 Enterprise Edition) para criação de servidor de aplicação. Um exemplo desta utilização é o servidor de aplicação JBoss. O servidor JBoss é um open source destinado a servir aplicações Java, é o terceiro mais utilizado, Linux e Apache são os dois primeiros. Em uma arquitetura web de três camadas, o servidor JBoss ocupa a segunda camada, onde controla a inteligência da aplicação organizando o tráfego de dados entre o servidor web e o banco de dados.

2.10. MySQL

O MySQL é um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL (Structured Query Language - Linguagem de Consulta Estruturada) como interface. É atualmente um dos bancos de dados mais populares, com mais de 4 milhões de instalações pelo mundo [WIKIPEDIA, 2006d].

2.10.1. Principais características

- Portabilidade (suporta praticamente qualquer plataforma atual) Compatibilidade (existem drivers ODBC, JDBC e .NET e módulos de interface para diversas linguagens de programação, como Java, C/C++, Python, Perl, PHP e Ruby)
- Excelente desempenho e estabilidade
- Pouco exigente quanto a recursos de hardware
- Facilidade de uso
- É um Software Livre
- Suporte a vários tipos de tabelas (como MyISAM e InnoDB), cada um específico para um fim.
- Poucos recursos quando comparados como outros banco de dados, como o PostgreSQL. Algumas deficiências (que os desenvolvedores do MySQL pretendem suprir): trigger, funções, dentre outros recursos.

2.10.2. Vantagens

Outra grande vantagem é a de ter código aberto e funcionar em, quase, qualquer plataforma e sistema operacional : Windows, Linux, FreeBSD, BSDI, Solaris, Mac OS X, SunOS, SGI, etc.

No passado, devido a não possuir (até a versão 3.x) funcionalidades consideradas essenciais em muitas áreas, como stored procedures, two-fase commit, subselects, foreign keys ou integridade referencial, é frequentemente considerado um sistema mais "leve" e para aplicações menos exigentes, sendo preterido por outros sistemas como o PostgreSQL[WIKIPEDIA, 2006d]..

2.10.3. Notas

O MySQL a partir da versão 4.1 adicionou suporte a Transações, SubSelects, Foreign Keys e Integridade Referencial. Esse suporte foi graças ao database engine InnoDB.

Com a versão 5.0, o MySQL incorporou mais recursos avançados ao sistema, incluindo views , triggers, storage procedures e transações XA.

2.10.3.1. Utilizações

- LiveJournal, 300 milhões de páginas vistas por dia.
- Amazon.com
- Slashdot, 50 milhões de páginas vistas por dia.
- Prêmio MySQL de Aplicações do Ano de 2005:
- CNET Networks
- Friendster, mais de 85 milhões de páginas vistas por dia, capaz de suportar mais de 1.5 bilhões de queries por dia.
- Wikipedia, mais de 200 milhões de queries por dia e 11.000 queries por segundo.

2.11. PORTA SERIAL

A porta serial é um dispositivo de E/S do computador onde os bits de dados são transmitidos um de cada vez em um único arquivo, ou seja de forma serial.

Trabalhando de forma bi-direcional, a porta serial pode se comunicar de forma síncrona ou assíncrona. No modo síncrono existe um canal de dados e um de sincronismo, alertando para os inícios e fins dos dados, enquanto o assíncrono o mesmo canal transmite estes sinais de controle de dados. Para isso são usados os sinalizadores conhecidos com *start bit* e *stop bit*, necessitando de um algoritmo no equipamento externo que interprete estes sinais e manipule os dados de forma adequada [WHITE, 1997].

O padrão atual para as comunicações seriais chama-se RS232, mas há muitas variações. Por exemplo, uma porta serial pode ter 9 ou 25 pinos.

2.11.1. RS232

RS é uma abreviação de “Recommended Standard ”, que em português significa Padrão Recomendado. Ela relata uma padronização de uma interface comum para comunicação de dados entre equipamentos, criada no início dos anos 60, por um comitê conhecido atualmente como “Electronic Industries Association” (EIA). Naquele tempo, a comunicação de dados compreendia a troca de dados digitais entre um computador central (mainframe) e terminais de computador remotos, ou entre dois terminais sem o envolvimento do computador. Estes dispositivos poderiam ser conectados através de linha telefônica, e conseqüentemente necessitavam um modem em cada lado para fazer a decodificação dos sinais [CANZIAN, 2005].

Dessas idéias nasceu o padrão RS232. Ele especifica as tensões, temporizações e funções dos sinais, um protocolo para troca de informações, e as conexões mecânicas. Há mais de 30 anos desde que essa padronização foi desenvolvida, a EIA publicou três modificações.

A mais recente, EIA232E, foi introduzida em 1991. Ao lado da mudança de nome de RS232 para EIA232, algumas linhas de sinais foram renomeadas e várias linhas novas foram definidas.

2.11.2. Taxa de Transferência (Braud Rate)

A taxa de transferência é a velocidade com que as informações (dados) são enviadas através de um canal, sendo medido em transições elétricas por segundo. Na norma EIA232, ocorre uma transição de sinal por bit, e a taxa de transferência e a taxa de bit (bit rate) são idênticas. Neste caso, uma taxa de 9600 bauds corresponde a uma transferência de 9600 dados por segundos, ou um período de aproximadamente, 104 μs (1/9600 s).

Outro conceito é a eficiência do canal de comunicação que é definido como o número de bits de informação utilizável (dados) enviados através do canal por segundo. Ele não inclui bits de sincronismo, formatação, e detecção de erro que podem ser adicionados à informação antes da mensagem ser transmitida, e sempre será no máximo igual a um.

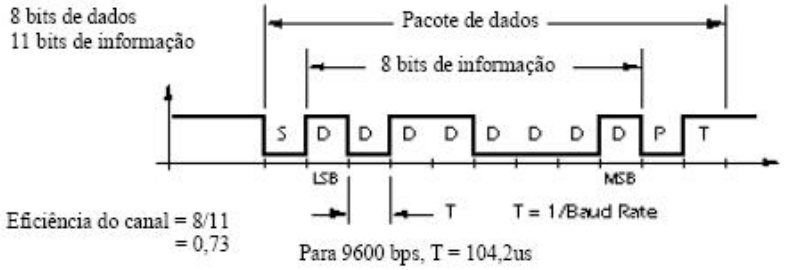

Figura 2.4 - Baud rate

2.11.3. Definição de sinais

Se a norma EIA232 completa for implementada, o equipamento que faz o processamento dos sinais é chamado DTE (Data Terminal Equipment – usualmente um computador ou terminal), tem um conector DB9 macho. O equipamento que faz a conexão (no caso do projeto será um PIC4F52) é denominado de DCE (Data Circuit-terminating Equipment), tem um conector DB9 fêmea. Um cabo de extensão entre dispositivos DTE e DCE contém ligações em paralelo, não necessitando mudanças na conexão de pinos. Se todos os dispositivos seguissem essa norma, todos os cabos seriam idênticos, e não haveria chances de haver conexões incorretas.

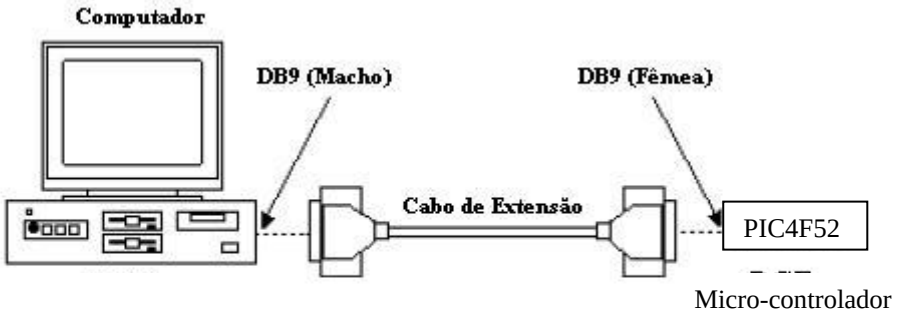

Figura 2.5 - Conexão DTE/DCE

2.11.4. Conectores

Na figura a seguir é apresentada à definição dos sinais para um dispositivo DTE (usualmente um micro PC). Os sinais mais comuns são apresentados em negrito.

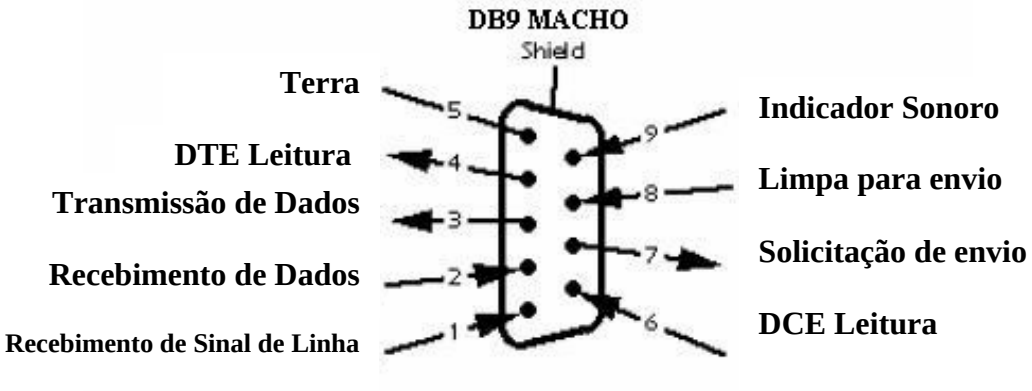

Figura 2.6 - DB9 – Macho

Na Figura 2.7 é apresentada a definição dos sinais para um dispositivo DCE. Os sinais mais comuns são apresentados em negrito.

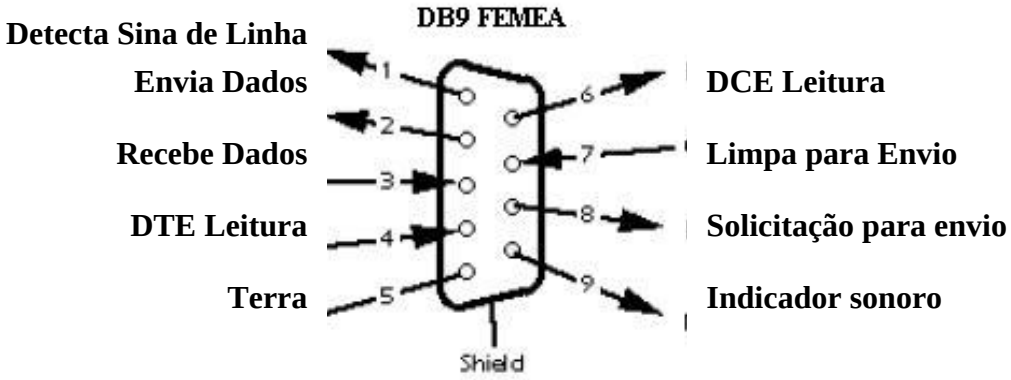

Figura 2.7 - DB9 – Fêmea

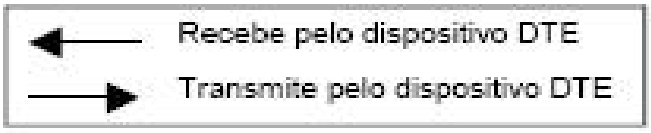

Figura 2.8 - Definições dos sinais do DB9 Macho/Fêmea

A Tabela 2.2apresenta a convenção utilizada para os sinais mais comuns do conector DB-9:

Tabela 2.2- Descrição dos pinos do conector DB9

Pino	Nome	Descrição
1	Carrier Detect (CD)	Também chamado de Data Carrier Detect (DCD). Este sinal é relevante quando o DCE for um modem. Ele é habilitado (nível lógico “0”) quando a linha telefônica está “fora do gancho”, uma conexão for estabelecida, e um tom de resposta começar a ser recebido do modem remoto. Este sinal é desabilitado (nível lógico “1”) quando não houver tom de resposta sendo recebido, ou quando o tom de resposta for de qualidade inadequada para o Modem local.
2	Received Data (RxD)	Este sinal está ativo quando o DTE receber dados do DCE. Quando o DCE estiver em repouso, o sinal é mantido na condição de marca (nível lógico “1”, tensão negativa).
3	Transmitted Data (TxD)	Este sinal está ativo quando dados estiverem sendo transmitidos do DTE para o DCE. Quando nenhum dado estiver sendo transmitido, o sinal é mantido na condição de marca (nível lógico “1”, tensão negativa).
4	DTE Ready (DTR)	Também chamado de Data Terminal Ready. Este sinal é habilitado (nível lógico “0”) pelo DTE quando for necessário abri o canal de comunicação. Se o DCE for um modem, a habilitação do sinal DTR prepara o modem para ser conectado ao circuito do telefone, e uma vez conectado, mantém a conexão. Quando o sinal DTR for desabilitado (nível lógico “1”), o modem muda para a condição “no gancho” e termina a conexão.
5	Ground (GND)	Sinal de terra utilizado como referência para outros sinais.
6	DCE Ready (DSR)	Também chamado de Data Set Ready. Quando originado de um modem, este sinal é habilitado (nível lógico “0”) quando as seguintes forem satisfeitas: 1 – O modem estiver conectado a uma linha telefônica ativa e “fora do gancho”; 2 – O modem estiver no modo dados; 3 – O modem tiver completado a discagem e está gerando um tom de resposta. Se a linha for tirada do gancho, uma condição de falha for detectada, ou uma conexão de voz for estabelecida, o sinal DSR é desabilitado (nível lógico “1”).
7	Request To Send (RTS)	Este sinal é habilitado (nível lógico “0”) para preparar o DCE para aceitar dados transmitidos pelo DTE. Esta preparação inclui a habilitação dos circuitos de recepção, ou a seleção a direção do canal em aplicações half-duplex. Quando o DCE estiver pronto, ele responde habilitando o sinal CTS.
8	Clear To Send (CTS)	Este sinal é habilitado (nível lógico “0”) pelo DCE para informar ao DTE que a transmissão pode começar. Os sinais RTS e CTS são comumente utilizados no controle do fluxo de dados em dispositivos DCE.
9	Ring Indicator (RI)	Este sinal é relevante quando o DCE for um modem, e é habilitado (nível lógico “0”) quando um sinal de chamada estiver sendo recebido na linha telefônica. A habilitação desse sinal terá aproximadamente a duração do tom de chamada, e será desabilitado entre os tons ou quando não houver tom de chamada presente.

Como pode ser observado neste capítulo, são muitas as aplicações da rede Internet e tecnologias de celulares no mundo dos negócios. Diversos softwares são utilizados no desenvolvimento dessas aplicações cada vez mais impulsionadas pelo crescimento rápido do comércio eletrônico. No próximo capítulo, temos a descrição da proposta de desenvolvimento e implementação de uma possível aplicação que possa ser utilizada para compras em máquina de auto-atendimento usando celular.

CAPÍTULO 3

IMPLEMENTAÇÃO DO PROJETO DE COMPRAS EM MÁQUINAS DE AUTO-ATENDIMENTO USANDO O CELULAR.

A implementação deste projeto foi realizada de modo a simular toda transação entre o cliente (que estará realizando a compra), o provedor de serviços de telecomunicações (responsável pela autorização da compra e cobrança do cliente) e a máquina de auto-atendimento que contem o produto a ser vendido.

Para tanto, a construção do projeto foi segmentada em três pontos:

- Especificação, desenvolvimento e simulação do servidor da operadora do celular;
- Especificação e desenvolvimento do Software para aparelho móvel;
- Especificação e simulação do dispositivo da máquina de auto-atendimento.

A seguir o fluxograma do projeto:

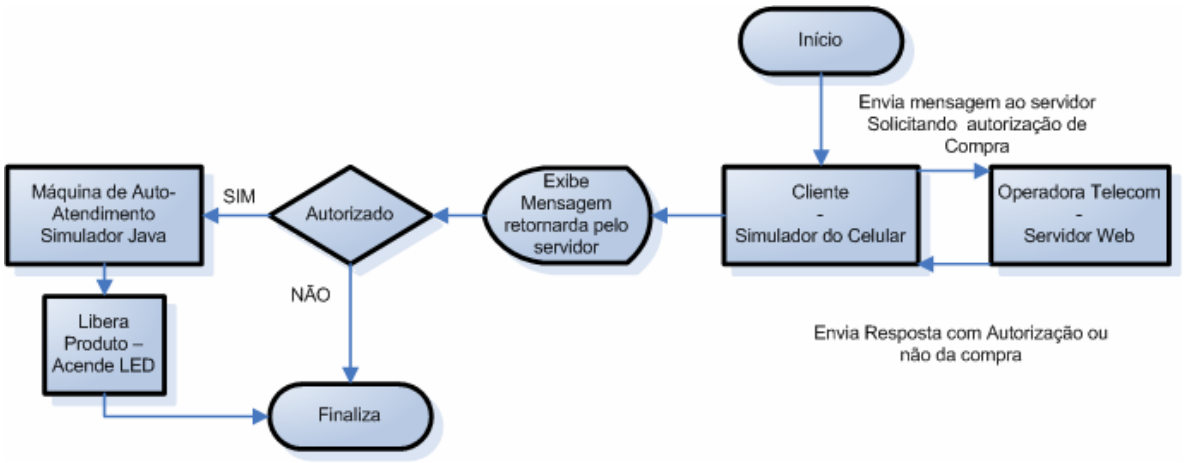

Figura 3.1– Fluxograma do Projeto

3.1. SIMULAÇÃO DO SERVIDOR DA PRESTADORA DE SERVIÇOS DE CELULAR

A primeira etapa de implementação do projeto trata da criação de um ambiente capaz de simular os procedimentos que seriam de responsabilidade da empresa prestadora de serviços de Telecomunicação.

De acordo com este projeto, são de responsabilidade da empresa de telecomunicação:

- Cadastro e controle de clientes (Nome, endereço, número do celular, situação financeira junto a prestadora);
- Prover serviços de envio e recebimento de mensagens SMS;
- Cadastro das máquinas de auto-atendimento e seus respectivos produtos;
- Cobrança dos serviços de dados, ou seja, a empresa seria responsável em cobrar do cliente o valor da compra realiza na máquina de auto-atendimento, seja na sua fatura telefônica (aparelhos pós-pagos), seja do seu crédito do celular (pré-pago);

Para o perfeito funcionamento do ambiente de simulação e testes do projeto foi construído um servidor web capaz de realizar as tarefas acima citadas.

3.1.1. Servidor de Banco de Dados

Foi construído um banco de dados capaz de armazenar as informações básicas contidas no banco de dados da operadora de telecom. Esses dados são válidos para a realização dos testes e simulações do processo descrito neste projeto.

Foi escolhido o sistema de gerenciamento de banco de dados MySQL. A escolha deste banco se deve ao fato de ser um banco que apesar de robusto, é leve, ou seja, ele

possui alta capacidade de armazenamento e de acesso simultâneo mas não sobrecarrega a máquina que o utiliza, ocupa pouco espaço em memória se comparado a outros.

Também foi considerado o fato de ser um banco amplamente utilizado para servir aplicações Java, como o caso deste projeto, e por ser um software livre, gratuito.

O banco deste sistema é constituído de duas tabelas. A primeira tabela tem o nome de “clientes” e armazena dados dos clientes como: nome, endereço, saldo, número do celular. A outra foi nomeada com “auto_atendimento”, responsável pelo cadastro das máquinas de auto-atendimento e contem: o código das máquinas, os produtos comercializáveis, preço e estoque.

Os registros da tabela “clientes” são identificados univocamente pelo número do aparelho celular, ou seja, um mesmo número de telefone não pode pertencer a dois clientes. Desta forma, a partir deste número é possível identificar o cliente e realizar as operações de consulta e alteração do mesmo.

Essa informação é importante pois o numero do celular é o parâmetro que será passado pelo telefone celular do usuário no momento da realização da compra para identificar o cliente no qual a compra será debitada.

A tabela “auto_atendimeto” identifica em qual máquina e produto está sendo vendido. A identificação é feita pelo campo código de máquina. Este campo contêm 6 caracteres, sendo os dois primeiros para identificar a máquina que esta sendo utilizada e os quatro últimos caracteres identificam o produto.

e realiza o controle de estoque. Lembrando que a existência ou não do produto no estoque é condição para realização da transação.

Esses dados serão utilizados pelo servidor WEB para analisar as condições da realização da compra ou não.

Os demais campos da tabela estão dispostos conforme esquema abaixo:

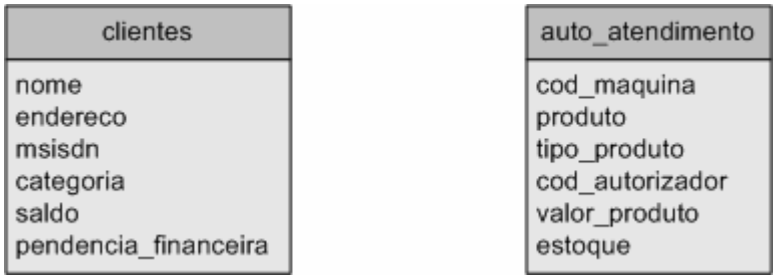

Figura 3.2 – Banco de dados de teste e simulações

Caso a compra seja autorizada o saldo do cliente será alterado no banco de dados, descontado o valor da compra assim como será alterado o estoque do produto.

3.1.2. Servidor Web

A prestadora de telefonia celular é responsável por receber a solicitação de compra enviada pelo cliente pelo seu aparelho celular, validar as condições do cliente e da máquina de auto-atendimento para realização do processo de compra.

Para realização destas funções foi construído um servidor web capaz de simular as funcionalidades da operadora de telecom que serão detalhadas neste tópico.

O servidor web foi construído utilizando o servidor de aplicações Apache Tomcat. Sua principal característica técnica é estar centrada na linguagem de programação Java, mais especificamente nas tecnologias de Servlets e de JSP (Java Server Pages), tecnologias utilizadas com base para o desenvolvimento deste projeto.

Para se desenvolver a aplicação web que será executada pelo Tomcat, para este projeto foi necessário o domínio das seguintes linguagens:

- Java – os algoritimos da aplicação devem estar escritos em Java;
- HTML – para criar páginas que serão visualizadas pelo browser do usuário;
- XML – para gerar os dados de configuração que podem ser usados tanto pelo Tomcat como pela aplicação.

Para a solução aqui implementada foi desenvolvida uma servlet “ServletServidorOperadora.java” para ser executada pelo Tomcat.

Essa Servlet permite a geração de conteúdo dinâmico e a interação com os clientes, utilizando o modelo request/response, ou seja, recebe a solicitação pelo celular do cliente e retorna o resultado da validação.

O Servlet criado é o cérebro da aplicação cliente servidor que utiliza JSP. A lógica está toda dentro das classes que compõem o container.

A “ServletServidorOperadora.java” que não é nada mais que uma classe java, é responsável por capturar a mensagem enviada pelo emulador do celular e tratar as informações recebidas e enviar a resposta da transação.

Para acionar a servlet é utilizado uma URL, ou, seja o endereço do servidor web conforme o padrão abaixo:

“http://servidor:porta_do_servidor_web/servidor_web/nome_servlet”.

Um exemplo de chamada para esse projeto seria:

http://localhost:8080/operadora/ServletServidorOperadora?MSISDN=556184101000&CODMAQ=000001”

Esta URL seria enviada pelo aparelho celular do cliente, em termos de projeto foi enviado pelo emulador do celular.

Ao receber a mensagem, a servlet vai separar os parâmetros recebidos: MSISND(Número do Celular) e CODMAQ(Código da Máquina de Auto-atendimento).

Com essa informação a servlet se utiliza de outros objetos do servidor web para realizar as validações necessárias como:

- ClienteBO.java – (BO - Business Object). Nele estão as regras de negócio do cliente como validação de saldo pra realizar a compra;

- MaquinaAtendimentoBO.java – Regra de negócio para as máquinas de auto-atendimento.Exemplo: Código de máquina válido, estoque de produto disponível;
- ClienteDAO.java – (DAO –Data Access Objetc) responsável por acessar informações sobre o cliente no banco de dados e armazenas no objeto(ClienteVO.java);
- MaquinaAtendimentoDAO.java - acessar informações sobre a maquina de auto-atendimento no banco de dados e armazenas no objeto(MaquinaAtendimentono VO.java)
- ClienteVO.java - (Value Obejct) – Contem atributos contento as informação recolhidas do banco de dados como nome, numero do telefone, etc.
- MaquinaAtendimentVO.java - Contem atributos contento as informação recolhidas do banco de dados como código autorizador de venda e estoque.

De acordo com o resultado das validações realizadas com estes objetos, será retornado pela servlet (ServletServidorOperadora.java) o código de autorização da compra ou a informação de que a compra não pode ser efetuada e sua justificativa. Os cenários simulações e testes serão apresentados no capítulo 4.

Este retorno será visualizado pelo emulador do celular.

3.2. SOFTWARE PARA APARELHO MÓVEL

De acordo com a proposta deste projeto, será por intermédio do aparelho celular do cliente que o mesmo solicitará autorização para realizar a retirada do produto na máquina de auto-atendimento.

Será utilizado um emulador de telefone celular para simular o aparelho do usuário, o WAC. Este programa tem como função simular o envio de mensagens SMS pela rede celular, no caso simulando a solicitação de compra e recebimento da autorização da transação.

Estas informações são enviadas para um servidor WEB que simula a prestadora de serviços que libera a compra. A forma de comunicação é síncrona e o emulador envia a mensagem para o servidor web.

Em uma implementação real (sem simulação) seria enviado um SMS para operadora que após validar os dados enviaria um outro SMS ao Cliente.

As informações que serão enviadas pelo celular são:

- Código da máquina de auto-atendimento digitado pelo usuário. Este código identificará no servidor em qual máquina a compra está sendo realizada.
- Número do telefone celular do Cliente. Essa informação é enviada automaticamente pela aplicação sem que o usuário tenha que digita-la. Com esse número o servidor validará se o cliente tem crédito para realizar a compra.

O uso do SMS é justificado pelo fato de ser uma tecnologia capaz de enviar curtas mensagens de texto (de até 160 caracteres) de forma rápida e fácil, o que se aplica perfeitamente à aplicação do projeto aqui proposto.

Outro fator relevante na escolha das mensagens SMS ou Torpedos como são usualmente conhecidos, é o fato de ser uma tecnologia de fácil manipulação e bastante difundida entre grande parte dos usuários de telefones celulares.

Vale ressaltar, que a segurança dos dados trafegados não é tratada com relevância neste projeto. O intuito deste é apenas realizar a comunicação entre as pontas (Cliente / Servidor).

3.2.1. WIG Application Creator (WAC)

Para o desenvolvimento do software para o aparelho celular, foi utilizado neste projeto a ferramenta WIG. Isso porque, ela permite simular aplicações de telefones móveis utilizando linguagem WML e ainda se comunica diretamente de um endereço URL no web Server, conforme descrito no capítulo 2, seção 2.5.

As escolha do WAC como ferramenta de desenvolvimento e testes se deve primeiramente ao fato de ser uma versão livre disponibilizada pela empresa SmartTrust.

O segundo motivo foi a facilidade para se desenvolver uma aplicação para celular se comparada a outras ferramentas que se utilizam de Java por exemplo.

O terceiro motivo, foi a constatação que esta ferramenta é utilizada hoje em grandes empresas de telecomunicação, o que certifica sua qualidade.

Em relação ao código desenvolvido para este projeto, pode-ser dizer que é um código simplório. A linguagem WML associadas ao WAC traz uma facilidade ao programador pois os comandos são escritos em tags. Estas tags são interpretadas pelo WAC que emula o celular já com as funcionalidades descritas no código fonte.

No caso específico deste projeto, foi construído um menu que pode ser visualizado na tela do aparelho celular emulado, contendo a opção de utilizar o serviço de compras em máquinas de auto-atendimento.

Ao escolher esta opção uma nova tela é mostrada solicitando que o usuário digite o código da máquina de auto-atendimento.

Após a digitação do código e o click no botão que simula o envio da mensagem SMS, é feita uma conexão via http com o servidor Web que simula a prestadora de serviços de telecomunicação. A URL contendo o endereço do servidor WEB é cadastrada dentro do código fonte WML pela tag “go href” com por exemplo:

http://localhost:8080/operadora/ServletServidorOperadora?MSISDN=556184101000&CODMAQ=000001

”

Onde o parâmetro MSIDN (número do celular) seria capturado do próprio SIMCard do cliente e o CODMAQ (código da máquina de auto-serviço) seria capturado após a digitação do usuário, conforme explicado anteriormente.

Após a conexão, é esperada a resposta do servidor Web que é exibida na tela do telefone celular emulado. Esta resposta pode ser o recebimento da mensagem com o código de autorização da compra, o que indicaria que o processo ocorreu com sucesso ou mensagem de erro do servidor.

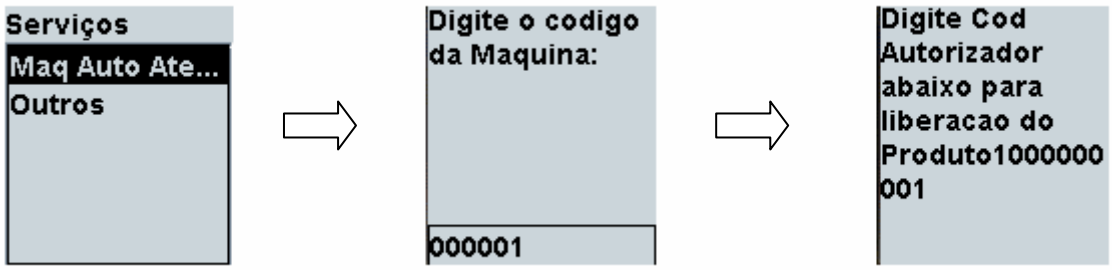

Figura 3.3 – Exemplo do visor do celular em caso de solicitação de compra realizada com sucesso.

Possíveis cenários desta simulação estão descritos no capítulo 4 deste trabalho.

3.3. SIMULAÇÃO DA MÁQUINA DE AUTO-ATENDIMENTO

Por fim, o último segmento para implementação do projeto é justamente simular por software a máquina de auto-atendimento.

Esta máquina, no caso este software, recebe manualmente pelo usuário o código autorizado retornado pelo servidor WEB para o seu celular para que o produto comprado seja liberado para o cliente.

Esse simulador é compostos por dois dispositivos:

- Software com interface para interação com o usuário (simula de forma gráfica a máquina de auto-atendimento);
- Placa PICLAB 4C com micro contralador 18F452 para simular os dispositivos eletrônicos das máquinas de auto-atendimento.

3.3.1. Simulação via Software

Foi construída uma aplicação Java para simular visualmente a máquina de auto-atendimento e realizar a interação com o micro-controlador.

Esta interface gráfica é constituída basicamente de um visor, um teclado, um visualizador do código da máquina, a opção de seleção de produtos e o botão de liberar produto.

De acordo com o produto selecionado um código de máquina e exibido no sistema. Isso para simular em um mesmo programa tipos de máquinas de auto-serviço diferente. Além disso, a composição deste código é capaz de identificar a máquina e o produto vendido.

Neste projeto não foram consideradas questões de segurança tal como a alteração do código de autorização Ou seja, cada produto, refrigerante, café, etc, possui um

único código autorizador. Esse código está cadastrado na banco de dados e também gravado nesta aplicação

Caso esse código seja informado corretamente ao programa será enviada uma mensagem pela porta seria de modo a informar ao micro-controlador qual o produto selecionado para simular a liberação do produto para comsumo.

Para a comunicação com a porta serial do computador, foi utilizada a API Java Comm. Com essa API o programador indica ao programa por qual porta ele deseja utilizar e o Java se responsabiliza pela comunicação. O código fonte deste sistema está disponível no Axeno B deste projeto.

3.3.2. Micro-Controlador 18F452

Para simulação dos dispositivos eletrônicos da máquina de auto-serviço e liberação do produto, foi utilizada a placa PICLAB 4B contendo um micro-controlador 18F452, comunicação serial e 8 LEDS.

A placa PICLAB 4B se comunica com o computador pela porta serial.

O programa Java que simula a máquina de auto-atendimento envia uma mensagem com 1 caracter pela porta serial do computador. Esse caracter é lido pelo micro-controlador que interpreta a informação e recebida e acende o LED que seria correspondente a liberação do produto selecionado.

Inicialmente o LED um fica piscando indicando que o programa está rodando.

Cada interrupção corresponderia a um tipo de produto diferente que deve ser liberado, então para cada produto cadastrado um LED diferente é acesso.

O acendimento do LED indica sucesso na operação caso contrário os 8 LEDS da máquina serão acessos.

A arquitetura da implementação esta distribuída conforme a figura a seguir:

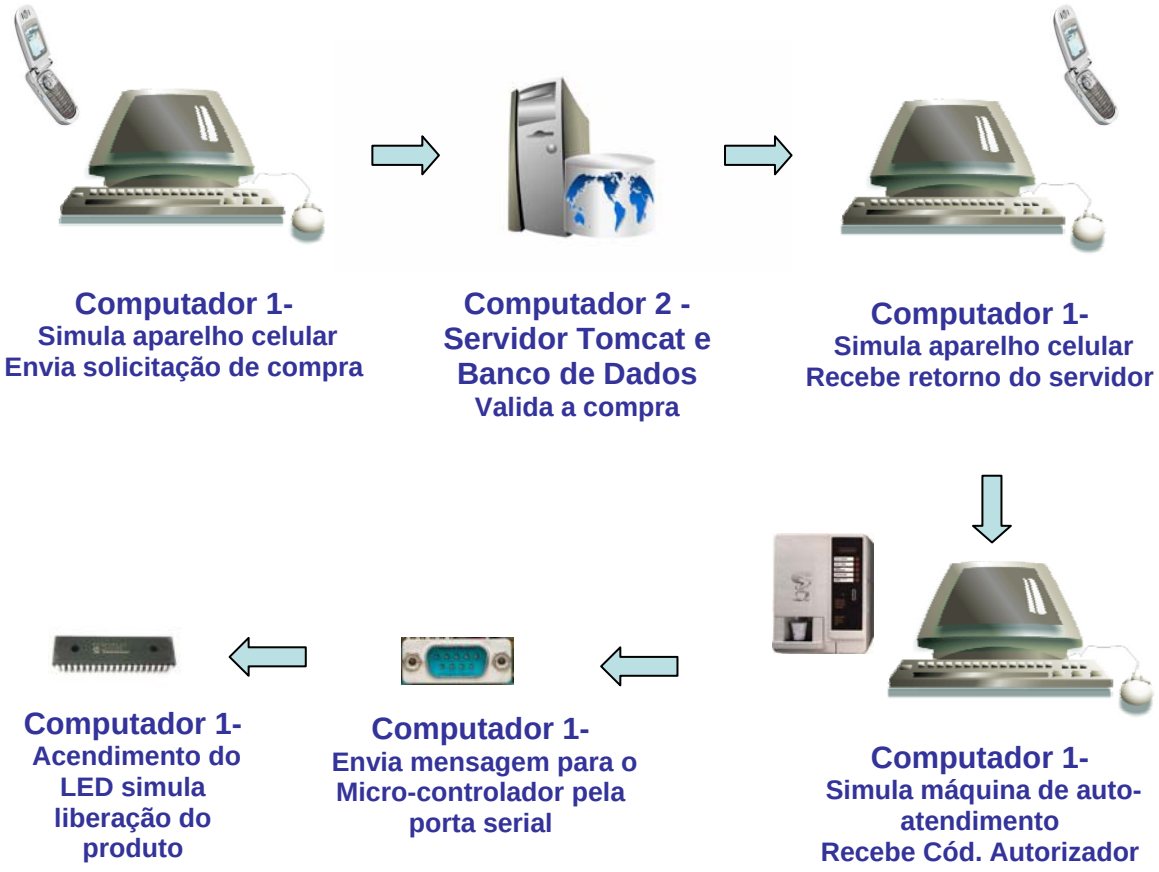

Figura 3.4 – Arquitetura de do Projeto

Com o auxílio das tecnologias descritas no capítulo 2 deste projeto foi possível desenvolver um sitesma capaz de realizar todas as simulações necessárias para as transações apresentadas neste projeto e detalhadas neste capítulo

No próximo capítulo estarão detalhados todos os procedimentos de simulação, testes e resultado colhidos após a implementação descrita no capítulo 3.

CAPÍTULO 4

SIMULAÇÕES E RESULTADOS

Este capítulo está destinado a exibir os resultados alcançados com as simulações e testes dos programas e componentes desenvolvidos e utilizados na implementação deste projeto.

Os testes foram realizados por etapas para cada um dos programas desenvolvidos.

Os programas foram separados em 3 grupos principais (conforme descrito no capítulo 3), cada grupo foi desenvolvido e testado conforme a ordem apresentada abaixo:

- Servidor (simulação do servidor da prestadora de serviços de telecomunicação);
    - Banco de Dados;
    - Servidor Web
- Programa para o aparelho celular (envia e recebe as mensagens de solicitação de compra);
- Máquina de auto-atendimento (Interface gráfica para simular máquina de auto-serviço que se comunicação com a porta serial do computador).

Cada grupo seguiu igualmente 3 etapas de teste:

- Teste unitário. Onde cada algoritmo de cada programa foi testado separadamente para assegurar os resultados das funções isoladamente.
- Teste de módulo. Cada um dos 3 grupos acima citados corresponde a um módulo, ou seja, um sistema (servidor da prestadora de serviços de telecomunicação, o celular e aMáquina de auto-atendimento). Nestes testes, todas as funções, ou algoritmos são testados em conjunto. A

entrada de dados externos é simulada de modo que apenas esse módulo seja testado. Desta forma, informações que deveriam vir de sistemas externos são simulados conforme a necessidade do teste.

- Teste integrado. A última etapa dos testes. Nesse momento a integração entre os módulos é testada, a comunicação entre os sistemas e a validação do conteúdo das informações passada.

O acompanhamento e resultado de todo esses testes são detalhado nos tópicos abaixo onde no primeiro momento será descrito os módulos separadamente (testes unitários e de módulo) e por fim o teste ponta a ponta de todo o projeto (teste integrado) onde serão descritos todos os cenários contemplados nas simulações.

4.1. SIMULAÇÃO DE TESTES UNITÁRIO E MÓDULO

4.1.1. Simulação do Servidor da prestadora de serviços de celular.

Conforme anteriormente relatado, a do servidor da operadora foi separado em banco de dados e servidor Web. Em cada fase da construção foram realizados testes.

4.1.1.1. Servidor Dados

Assim com a criação do banco de dados e a inserção dos dados, os primeiros testes no servidor de dados foram realizado via interface gráfica disponibilizada pelo MySQL.

Estes testes foram compostos de procedimentos de inserção de registro (INSERT), alteração (UPDATE) e remoção (DELETE), testando assim simples consulta a base de dados via query no formato SQL(SELECT).

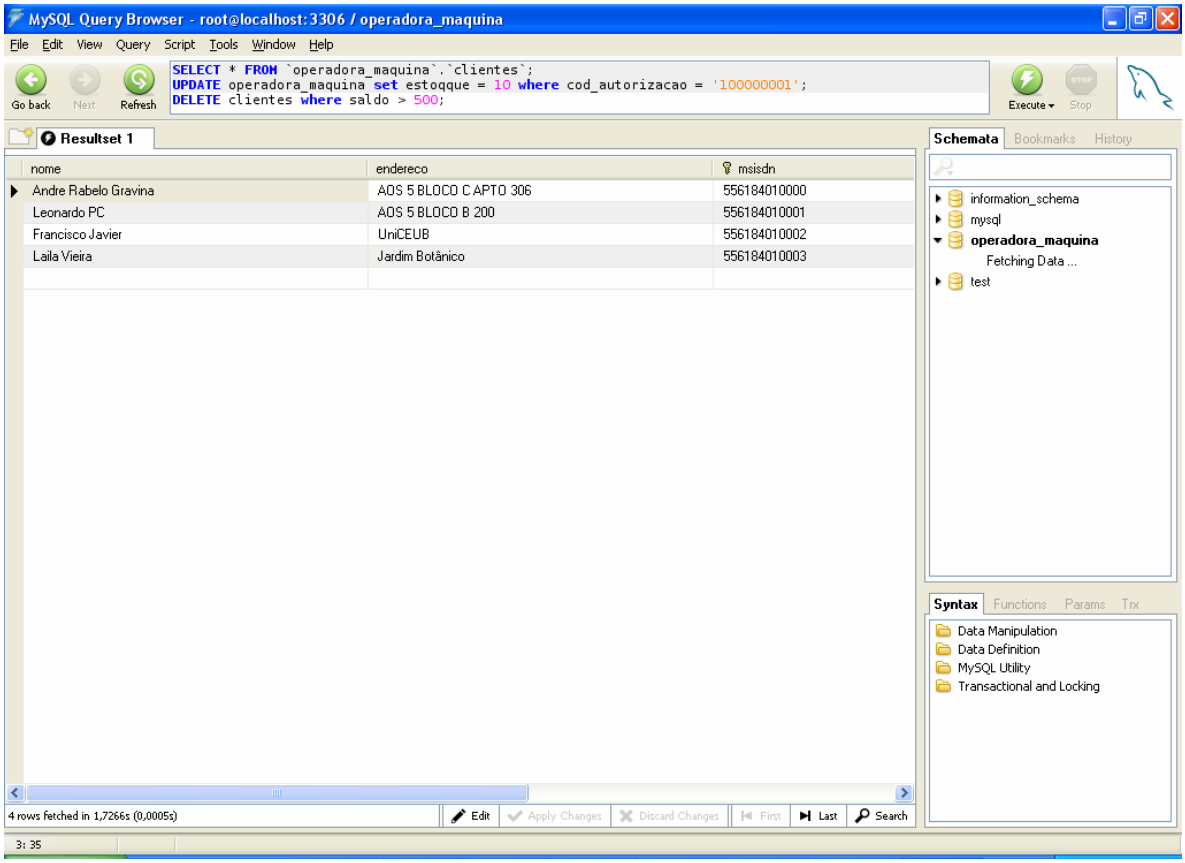

Figura 4.1 – Tela de login da ferramenta gráfica do MySQL. Usada para inserir, alterar e apagar registros no Banco de Dados.

4.1.1.2. Servidor Web

O servidor Web, composto pelo Apache Tomcat e pela Servle, é responsável por receber as mensagens de solicitação de compra, tratar os dados recebidos e eviar uma resposta para quem fez a chamada à servlet.

Para isso foi utilizada uma IDE (Integrated Development Enviroment), conforme [WIKPEDIA, 2006e], um programa de computador com características e ferramentas de apoio ao desenvolvimento de software, com o objetivo de agilizar este processo. A IDE utilizada foi o Eclipse SDK, versão 3.1.2, ferramenta gratuita que disponibiliza suporte para desenvolvimento e teste de aplicações Java. Além disso, ela fornece meios de interação com o Tomcat e Banco de Dados por intermédio de “plug in”.

O servidor Web, também foi estado conforme estava sendo construído. Os primeiros testes unitários foram com relação ao recebimento de dados para servlet. Ou seja, o servidor tomcat foi inicializado interagindo diretamente com a servlet. Os testes foram realizados enviando para o servidor apache uma mensagem digitada no browser do computador conforme exemplo:

http://localhost:8080/operadora/ServletServidorOperadora?MSISDN=556184101000&CODMAQ=000001"

Onde "localhost:8080" é o endereço e porta padrão do apache, "operadora" o projeto Java do servidor Web e "ServletServidorOperadora" a servlet testada.

Conforme o exemplo em estudo, a servlet deveria perceber a entrada dos dois parâmetros, MSISDN e CODMAQ e seus respectivos valores "556184101000" e "000001".

A partir de então, os demais objetos para realizar as validações do cliente e da máquina de auto-atendimento utilizam o acesso ao banco de dados para consulta ou alterações quando necessário.

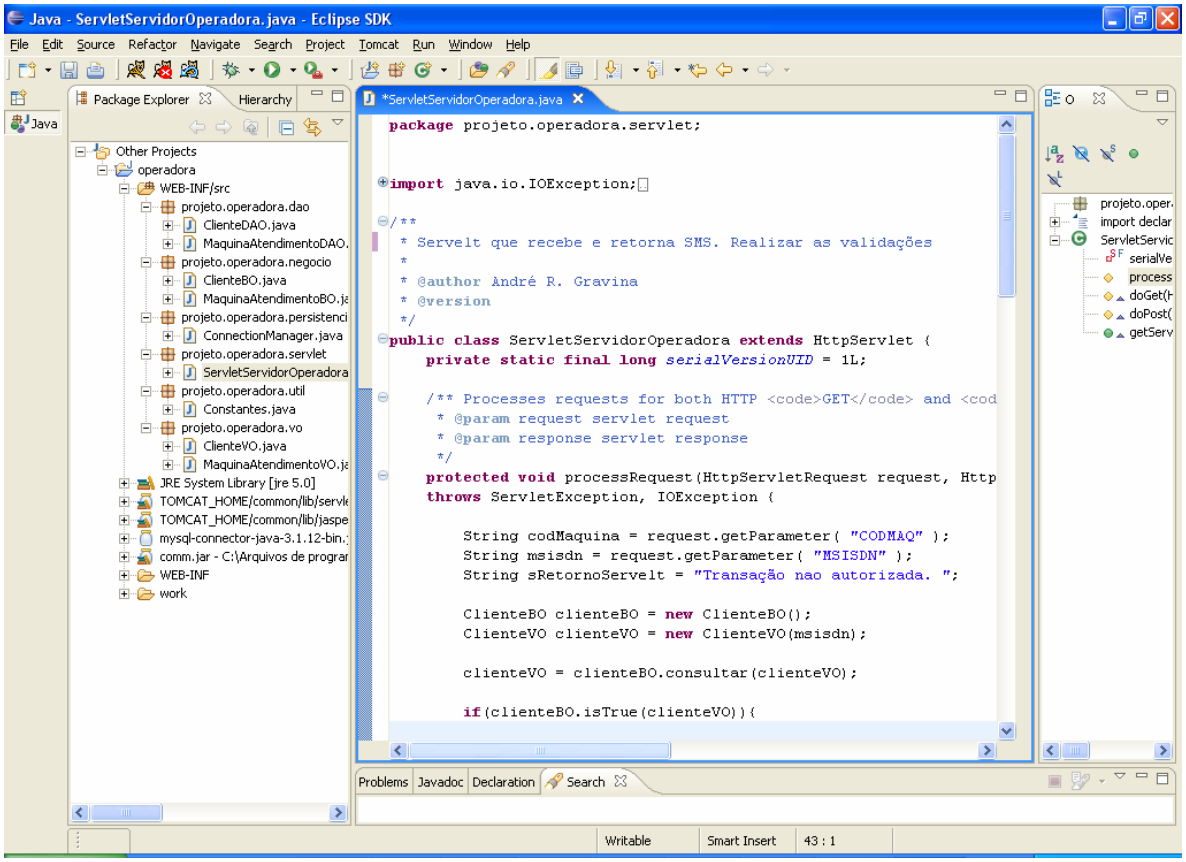

Figura 4.2 – Eclipe - Utilizada para o desenvolvimento do Servidor Web

4.1.2. Simulação do Aparelho celular

Para a construção, simulação e testes do programa elaborado para o envio de SMS pelo aparelho celular, foi utilizado um programa para desenvolviento de código e emulação de aparelho celular chamado WIG Application Creator, de propriedade da SmartTrust conforme detalhado no item 2.5 do Capítulo 2 de referencial teórico.

Os testes unitário e módulo realizados foram basicamente para criação do menu do celular, chamar o serviço de envio de SMS e o teste de envio e recebimento da mensagem, passando para o programa a URL com o endereço do servidor que deve receber a notificação.

Neste programa é cadastrado a URL com o endereço da servlet que deve receber a mensagem e retornar a resposta das validações.

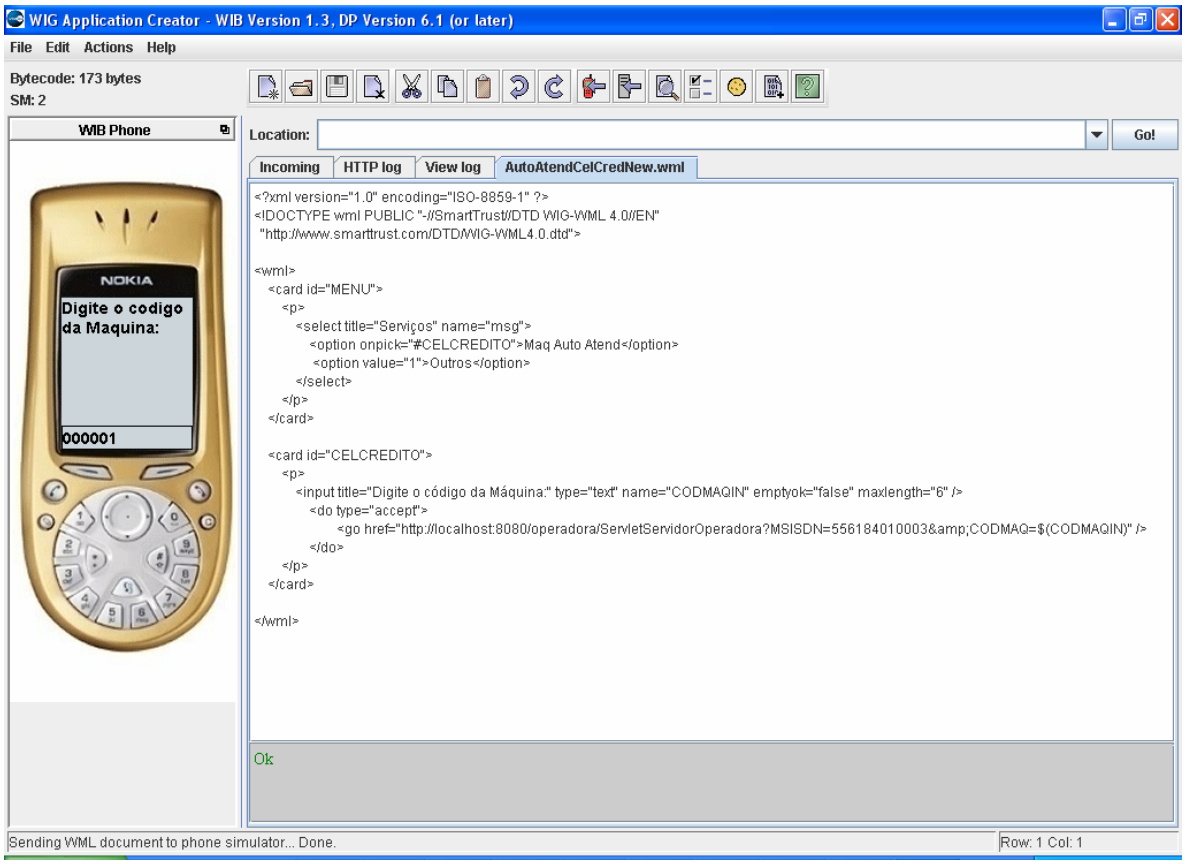

Figura 4.3 – WIG – Utilizada para o desenvolvimento do Software para o celular.

4.1.3. Simulação da Máquina de auto-atendimento

Por fim, foi implementado e testado um sistema, ou melhor um protótipo, capaz de simular o funcionamento da máquina de auto-atendimento neste projeto.

Esta etapa foi subdividida em duas partes

- Micro-controlador (simula o dispositivo eletrônico da maquina responsável pela liberação do produto).
- Interface gráfica Java (simula a interação com usuário);

4.1.3.1. Micro-controlador.

A primeira etapa do desenvolvimento do simulador da máquina de auto-atendimento está relacionada ao micro-controlador que recebe as informações da porta serial e realiza funções a partir disto.

O trabalho do micro-controlador é justamente para simular o que ocorre na pratica dentro de uma máquina de auto-atendimento, mas ao invés de liberar o produto para o consumo do cliente, o micro-controlador será responsável pelo acendimento de alguns LEDS de acordo com o produto selecionado pelo usuário.

Foi utilizada a placa PICLAB 4B, que já possui integrada a comunicação com a porta paralela (por onde o código é enviado para o micro-controlador), porta serial (recebe o sinal do software e envia para o micro-controlador), 8 LEDS (que simulam os produtos disponíveis na máquina) e o micro-controlador 18F452 que recebe programação C compilada em linguagem hexadecimal.

Para construír o código foi utilizado o bloco de notas do windows, mas para enviá-lo para micro-controlador era necessário antes converte-lo para hexadecimal.

Para esta finalidade foi usado o programa PWC compiler.

Para aplicação do código no controlador foi utilizado o programa WINPIC800.

Para testar apenas o código do micro-controlador (teste unitário) foi utilizado o progama RComSerial versão 1.1a [ROGERCOM, 2006]. Esse programa é capaz de enviar dados para a porta serial do computador, interagindo com o micro-controlador.

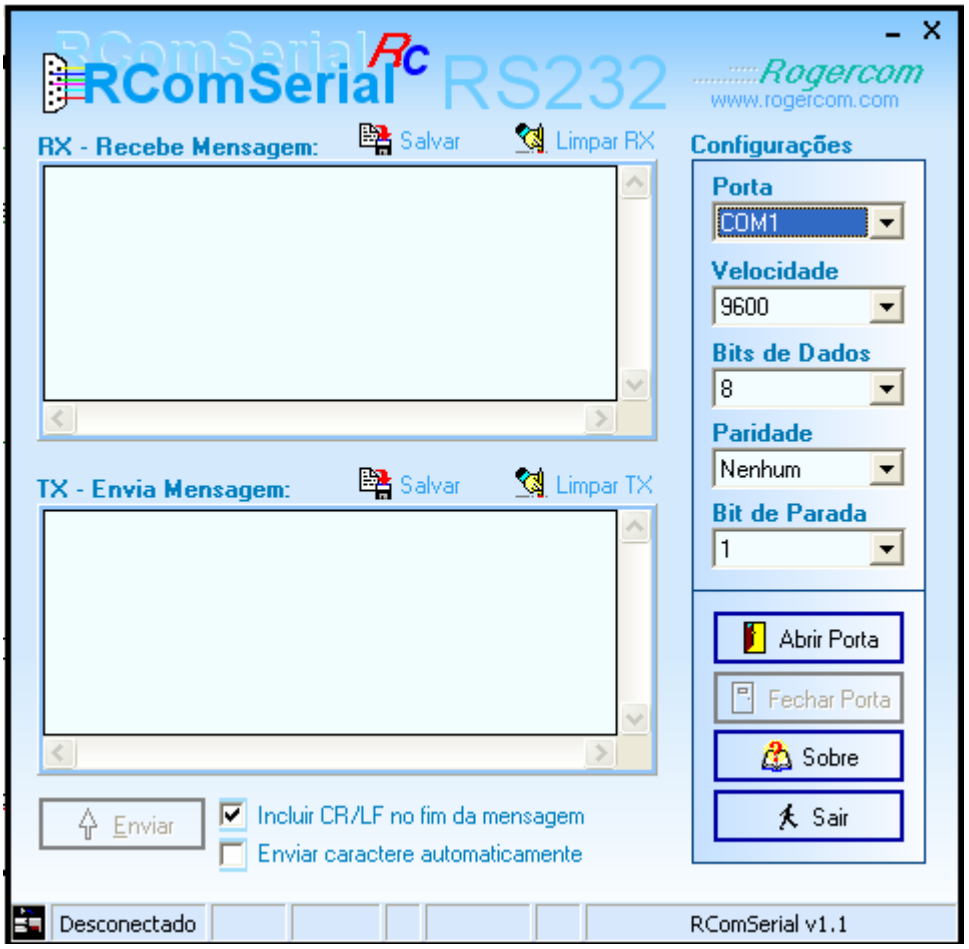

Figura 4.4– RComSerial – Envia dados pela porta serial

4.1.3.2. Interface gráfica

Para desenvolvimento desta interface foi escolhido a linguagem Java pela sua alta aceitação no mercado e pelo fato de possuir uma API, Application Programming Interface (ou Interface de Programação de Aplicativos) que um conjunto de rotinas e padrões estabelecidos pela proprietária do Java [WIKPEDIA, 2006], a SUN, que facilita a comunicação entre o software e a porta serial do computador e o dispositivo ligado a ela, no nosso caso um micro-controlador.

Neste caso, a IDE escolhida para desenvolvimento e testes deste programa foi o NetBeans 5.0. A escolha desta ferramenta se justifica pelo fato do NetBeans possuir uma enorme gama de ferramenta que auxiliam a construção de interfaces gráficas java.

Este programa visa simular as característica do componente que deve ser acrescido nas máquinas de auto-atendimento, ou seja, um teclado e uma visor para entrada da informação do código de autorização da compra. O mesmo sistema é responsável pela comunicação com a porta serial do computador.

No primeiro momento foi testado a organização da tela, e controle dos eventos como clique dos botões, escolha dos produtos e exibição das informações na tela.

A Figura 4.5 exibe a ferramenta Netbeans e parte do código da interface gráfica da máquina de auto-atendimento.

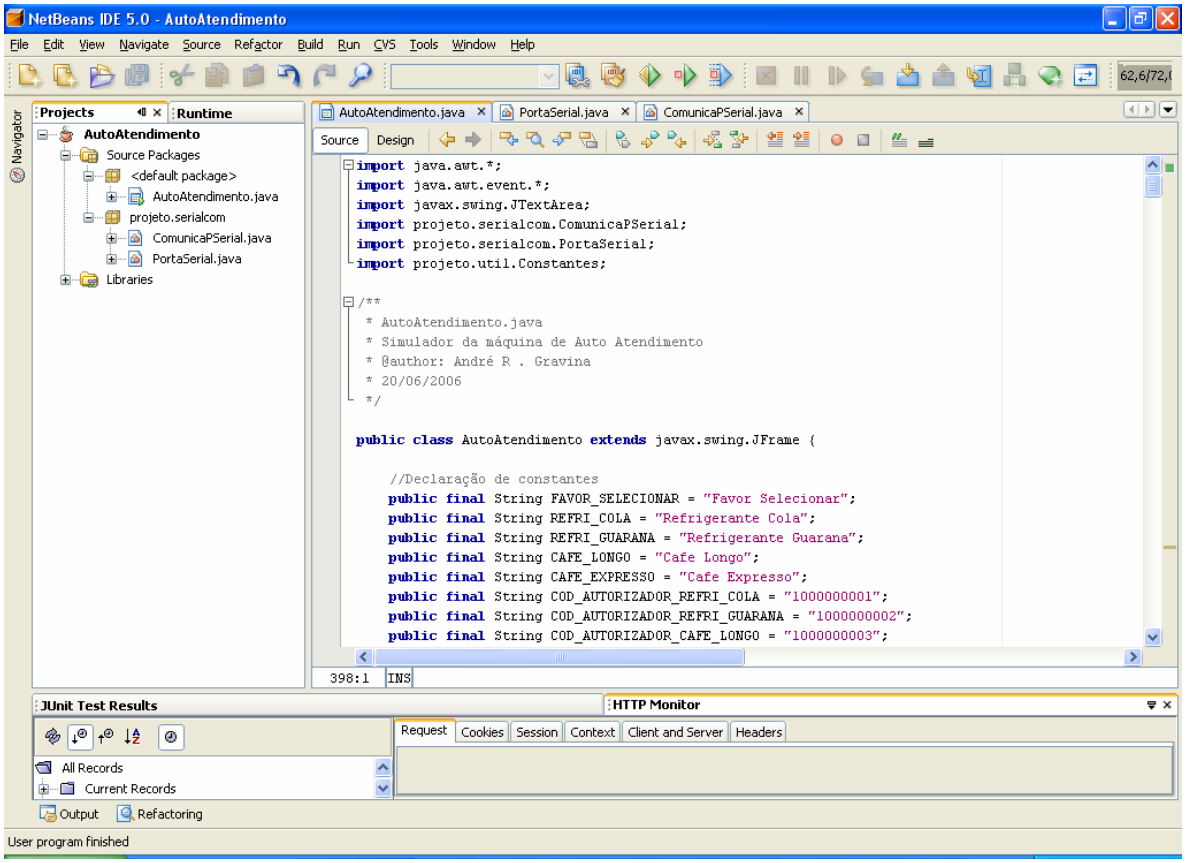

Figura 4.5 – Netbeans – Construção Interface Gráfica da Máquina e comunicação com a porta serial do microcomputador.

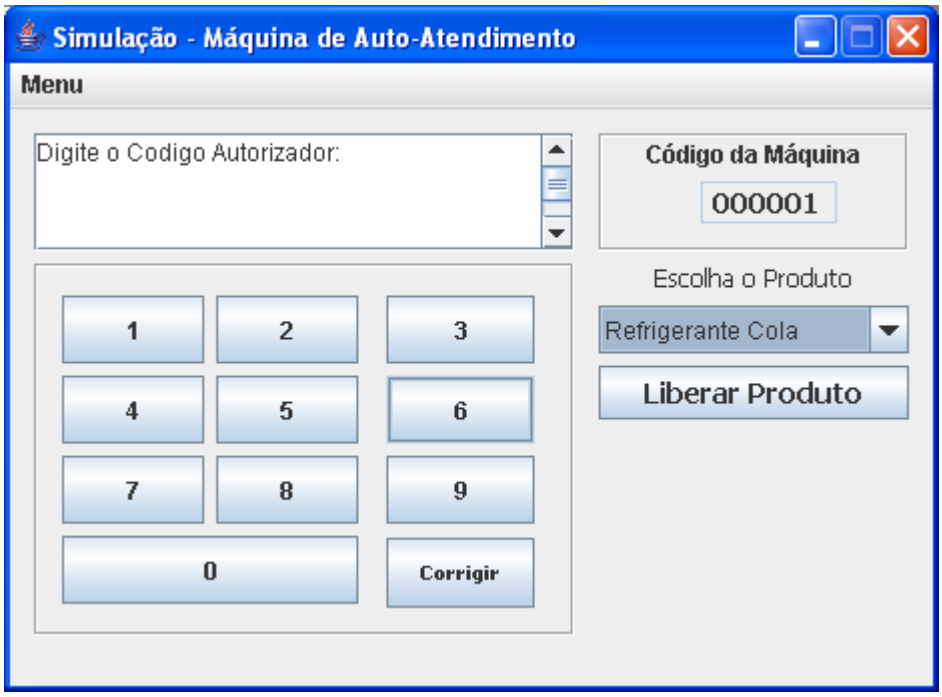

Figura 4.6 – Interface Gráfica do Programa.

No segundo momento a interface foi alterada e testada de modo que ao receber um código autorizador testasse se o código era o referente ao produto selecionado e envia-se para porta serial do computador a informação referente a liberação do produto.

Esse seria o primeiro processo de integração entre dois equipamentos nesse projeto, a comunicação entre a placa PIC e o software Java.

4.2. SIMULAÇÃO DE TESTE INTEGRADO

O teste integrado é a efetivação da simulação do processo ponta a ponta.

Estes testes também foram segmentados para que obtivéssemos uma maior segurança nos cenários propostos pelo projeto.

A primeira parte da integração ocorreu entre o celular e o servidor da prestadora de serviços, ou seja, entre o emulador do celular e o servidor web. A segunda parte foi entre o software da máquina de auto-serviço e a placa PIC.

4.2.1. Comunicação entre o telefone celular e o servidor.

Estes testes foram realizados no momento em que o cliente solicita o código de autorização para compra, o servidor valida as informações enviadas pelo celular e retorna o resultado da validação.

Os cenários testados foram:

Tabela 4.1 – Cenários do teste integrado Celular / servidor.

Celular / Servidor								
cenário	Celular			Cliente			Maquina / Produto	Venda Liberada
	Envio Cod Máquina	Nº Celular	Resposta do Servidor	Saldo	Pendência Financeira	Categoria	Estoque do produto	
1	válido	válido	código autorizador	N/V	Não	Pos-pago	com estoque	**ok**
2	válido	**inválido**	Falha no cadastro do Cliente	N/V	Não	Pos-pago	com estoque	Nok
3	**inválido**	válido	Código de máquina não encontrado	N/V	Não	Pos-pago	com estoque	Nok
5	válido	válido	Produto sem estoque disponível	N/V	Não	Pos-pago	**sem estoque**	Nok
6	válido	válido	Cliente com Pendencia Financeira	N/V	**Sim**	Pos-pago	com estoque	Nok
7	válido	válido	código autorizador	Suficiente	N/A	Pre-pago	com estoque	**ok**
8	válido	válido	Falha no cadastro do Cliente	**Não Suficiente**	N/A	Pre-pago	com estoque	Nok

N/V – Não verificado. Informação não necessária para o cenário
OK – venda realizada com sucesso
NOK – venda não realizada com sucesso

A tabela 4.1 contém todas as mensagens possíveis retornadas pelo servidor web e exibidas na tela do emulador do celular.

Para completo entendimento da tabela é válido considerar:

- **Código de máquina**: Inválido - O cliente pode ter digitado o número incorretamente ou a máquina não esta cadastrada no banco de dados do servido;

- **Número do Celular**: Inválido – Considera-se que o cliente não está cadastrado no banco de dados para realizar esse tipo de compra. Não é considerado que esteja sendo enviado um número de celular inválido pois o mesmo é obtido diretamente do SimCard do cliente, ou seja, do próprio chip GSM. Desta maneira descartamos a possibilidade de erro na digitação.
- **Celular Pós-pago:** Seu saldo não é levado em consideração pois a cobrança seria realizada na sua fatura telefônica. Não é permitida venda em caso de pendência financeira (fator considerado em caso de inadimplência com a prestadora de serviço por exemplo) .
- **Celular Pré-pago:** Valida se o saldo do cliente é maior que o preço do produto requerido.
- **Estoque:** Verifica estoque do produto para liberação da venda.

4.2.2. Software Máquina Auto-Atendimento / Micro-controlador

Como todas as validações sobre o código autorizador são realizadas pelo software Java, os testes de integração entre ele e a placa PIC são basicamente o envio via software para a porta serial uma mensagem para o micro-controlador para que possa interpretar qual produto deverá ser liberado, ou no caso da simulação, qual LED deverá ser acesso.

Para isso o programa Java da aplicação e o programa C no microcontrolador devem estar coerentes em relação aos produtos testados que foram 4:

- Refrigerante Cola – acende LED 2
- Refrigerante Guarana - acende LED 4
- Café Longo - acende LED 6
- Café Expresso - acende LED 8

- Em caso de erro – Acende todos os LEDs

É prudente lembrar que esses mesmos produtos devem estar cadastrados no banco de dados do servidor para que ao receber o código de máquina possa ser retornado o código autorizador referente ao produto solicitado.

Logo, o cadastramento deve estar alinhado entre Banco de Dados, Programa Java e Micro-controlador.

Essa metodologia de teste foi bastante importante e eficaz. Devido a dedicação aos teste das simulações foi possível descobrir uma série de erros não percebido no momento da codificação.

A seriedade dos testes culmina no aumento da qualidade do resultado final do trabalho esperado em um projeto de conclusão do curso de engenharia.

CAPÍTULO 5

CONCLUSÃO

Este projeto apresentou uma proposta de uma nova utilização para aparelho celular, onde ele é uma ferramenta para realizar compras em máquinas de auto-atendimento para aquisição de produtos como refrigerante, café, doces, salgados, etc.

Desta forma o cliente realiza a compra sem a necessidade de usar dinheiro, já que o custo da compra será debitado ou abatido do crédito do seu celular pré-pago.

Para o desenvolvimento deste projeto foi necessário um intenso trabalho de pesquisa e estudos tanto sobre as tecnologias necessárias para o seu desenvolvimento quanto a compressão da conjuntura atual do comércio eletrônico, principalmente no que se refere aos negócios realizados via celular (m-business).

Dentro do estudo teórico, houve a preocupação de identificar dentre as tecnologias quais seriam mais adequadas para serem utilizadas em relação a: sua utilização no mercado, praticidade para desenvolvimento de software e compatibilidade entre os sistemas envolvidos no processo.

Levando em consideração estes aspectos, foram escolhidas alternativas como:

- Celulares com tecnologias como o GSM, que atualmente abrange a maioria dos usuários de celular em termos globais e nacionalmente.
- SMS para envio de mensagens.
- Desenvolvimento dos softwares utilizando programas gratuitos e de alta qualidade e aceitação no mercado como Java, MySQL e Apache TomCat.

Devido ao caráter acadêmico deste projeto, algumas limitações práticas para implantação comercial foram substituídas por simulações.

Faz-se necessário, uma parceria entre a prestadora de serviços de telecomunicação e o fornecedor do serviço de máquina de auto-atendimento, pois as prestadoras seriam responsáveis pelo controle das vendas e pela cobrança dos clientes.

Além disso, não se teve acesso á tecnologia utilizada nas máquinas de auto-atendimento. Para suprir estas limitações foram construídos softwares capazes de simular de forma satisfatória as operações realizadas tanto pela prestadora de serviços de telecomunicação de celular, quanto pela máquina de auto-atendimento.

Com isso, foi possível demonstrar de forma prática o processo proposto neste projeto de graduação e ainda, comprovar a viabilidade tecnológica de se implementar no mercado um processo semelhante a este que possa de fato realizar compras em máquina de auto-atendimento utilizando o aparelho celular.

5.1. PROJETOS FUTUROS

Conforme as premissas deste projeto, ele apresenta características predominantemente acadêmicos, já que para sua implementação real e utilização comercial algumas limitaçõoes devem ser superadas.

Como propostas para projetos futuros são citados exatamente pontos que limitaram o uso comercial dessa idéia como:

- Controle de seguranças das informações trafegadas pela rede wireless. Ou seja, garantir a integridade dos dados que são enviados e recebidos pelo celular;
- Comunicação direta da máquina de auto-atendimento com o servidor da prestadora de serviços de telecomunização. Com isso o código de autorização seria enviado diretamente para a máquina de auto-atendimento, evitando problemas como fraude na obtenção do código e acesso e confirmação que o produto foi realmente liberado para o

- Desenvolvimento de uma aplicação capaz de realizar o controle das máquina de auto-atendimento via Internet. Como estoque e faturamento.
- Projeto / proposta para que o usuário possa consultar suas movimentações e compras realizadas pelo aparelho celular usando a Internet ou o recebimento de mensagens no seu próprio celular.

REFERÊNCIAS BIBLIOGRÁFICAS

ALEXANDRINE, Fábio. Perfil empresarial na prática do E-Commerce - comercialização eletrônica. Dissertação (Mestrado em Engenharia de Produção) - Programa de Pós-Graduação em Engenharia de Produção. Florianópolis: UFSC, 2000.

ALMEIDA, L. C. de P. O Comércio, a Internet e os Organismos Internacionais: construindo a estrutura do comércio eletrônico. Rio de Janeiro: CNC, 1999.

C. **Bettstetter**, H. Vogel, and J. Eberspacher, "GSM Phase 2+ General Packet Radio Service GPRS: Architecture, Protocols, and Air Interface," IEEE Communications Surveys, Vol. 2, No. 3, 1999,

INTEL. Enabling next generation e-business architectures: balancing integration and flexibility for managng business transformation. E-Business Strategy White Paper. BRINT Institute, Jun. 2001.
http://www.brint.org/intelebusiness.pdf - acessado em maio de 2006 não incluir na monografia

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

LAZILHA, F. R. E-business: negócios eletrônicos, estratégia empresarial. Revista Iniciação Científica/Núcleo de Iniciação Científica – NIC, CESUMAR, Maringá, v.1, n.1, p.31-42, ago./dez. 1999.

MELO, Marco Antonio. O comércio eletrônico e as novas formas de informação: do livro convencional ao livro eletrônico (e-book). Dissertação (Mestrado em Engenharia de Produção) - Programa de Pós-Graduação em Engenharia de Produção. Florianópolis: UFSC, 2002.

MENDES, A. A.; KANECO, P. A.; SOUZA, A. A. de; CRUBELLATE, J. M. O Comércio Eletrônico como estratégia de marketing e algumas considerações sobre o seu uso por uma empresa maringaense. Anais. XIX ENEGEP, Rio de Janeiro, 1999.

MOSKORZ, Rafael Roberto. M-COMMERCE: estratégias para difusão e implantação. Dissertação (Mestrado em Engenharia de Produção) - Programa de Pós-Graduação em Engenharia de Produção. Florianópolis: UFSC, 2002.

NASCIMENTO, Juarez do, Telecomunicações. Editora MAKRON Books Ltda 2000
RNP, Rede Nacional de Pesquisa. Popularização da Internet: introdução ao uso de correio eletrônico e Web. Instituto Tamis. Documento nr RNP/ref/0186. Rio de Janeiro. 1997.

SEYBOLD, Patricia; MARSHAK, Ronni T.. Clientes.c om. São Paulo: Makron Books, 2000.

SIEGEL, D. Futurize sua Empresa. São Paulo, Futura, 2000.

WATKINS, D, Overview and Comparison of GSM, GPRS and UMTS.
Bradley Department of Electrical and Computer Engineering,
Virginia Polytechnic Institute and State University, April 2000

WHITE, Ron. **COMO FUNCIONA O COMPUTADOR III**. Editora Quark, 1997.

Sites Citados

ANATEL, "**relatório anual anatel 2005**" http://www.anatel.gov.br/indicadores/default.asp Acessado em 29/06/2006

CAMARA,"**Varejo Online supera expectativa e fecha o ano em R$ 9,9 bilhões**" http://www.camara-e.net/interna.asp?tipo=1&valor=3505 Acessado em 06/06/2006

INFO, "**Internet tem 1 bilhão de usuários"**
http://info.abril.com.br/aberto/infonews/052006/19052006-2.shl Acessado em 06/06/2006

GSM, http://www.gsmworld.com/ Acessado em 08/04/2006

ROGERCOM, http://www.rogercom.com/ Acessado em 05/05/2006

SMARTTRUST, http://www.smarttrust.com/ Acessado em 20/01/2006

WIKIPEDIA, "**wireless**" http://pt.wikipedia.org/wiki/Wireless Acessado em 25/03/2006

WIKIPEDIA, "**java**" http://pt.wikipedia.org/wiki/Java Acessado em 18/05/2006

WIKIPEDIA, "**tomcat**" http://pt.wikipedia.org/wiki/tomcat Acessado em 18/05/2006

WIKIPEDIA, "**mysql**" http://pt.wikipedia.org/wiki/MYSQL Acessado em 18/05/2006

WIKIPEDIA, "**IDE**" http://pt.wikipedia.org/wiki/IDE Acessado em 20/05/2006

WIRELESSBRASIL, **"histórico do sistema móvel celular"**
http://www.wirelessbrasil.org/wirelessbr/secoes/sec_telefonia.html Acessado em 06/02/2006

WMW, http://www.wmw.com.br Acessado em 16/11/2005

Sites consultados

http://browser.netscape.com/ns8/ Acessado em 06/06/2006
http://pt.wikipedia.org Acessado em 25/03/2006
http://www.etsi.org/ Acessado em 08/04/2006
http://www.e-commerce.org.br/STATS.htm#D Acessado em
http://www.forum.nokia.com Acessado em 03/08/2005
http://www.guj.com.br/content/articles/netbeans Acessado em 03/05/2006
http://www.itu.int/ Acessado em 13/06/2006

http://www.microsoft.com/windows/ie/ie7/about/default.mspx Acessado em 06/06/2006
http://www.mhavila.com.br/topicos/java/tomcat.html Acessado em 03/05/2006
http://www.netbeans.org/kb/41/exploringmvd.html. Acessado em 06/04/2006
http://www.nic.br/indicadores/usuarios/index-ipsos.htm#c Acessado em 06/06/2006
http://www.rnp.br Acessado em 31/05/2006
http://www.3gpp.org/ Acessado em 15/09/2005

ANEXOS

Anexo A. CÓDIGO FONTE DO SERVIDOR WEB

ServletServidorOperadora.java

```
package projeto.operadora.servlet;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import projeto.operadora.negocio.ClienteBO;
import projeto.operadora.negocio.MaquinaAtendimentoBO;
import projeto.operadora.vo.ClienteVO;
import projeto.operadora.vo.MaquinaAtendimentoVO;

/**
 * Servelt que recebe e retorna SMS. Realizar as validações
 *
 * @author André R. Gravina
 * @version
 */
public class ServletServidorOperadora extends HttpServlet {
	private static final long serialVersionUID = 1L;

	/** Processes requests for both HTTP <code>GET</code> and <code>POST</code> methods.
    * @param request servlet request
    * @param response servlet response
    */
   protected void processRequest(HttpServletRequest request, HttpServletResponse response)
   throws ServletException, IOException {

	String codMaquina = request.getParameter( "CODMAQ" );
	String msisdn = request.getParameter( "MSISDN" );
	String sRetornoServelt = "Transação nao autorizada. ";

	   ClienteBO clienteBO = new ClienteBO();
	   ClienteVO clienteVO = new ClienteVO(msisdn);

	   clienteVO = clienteBO.consultar(clienteVO);

	   if(clienteBO.isTrue(clienteVO)){

		MaquinaAtendimentoBO maquinaAtendimentoBO = new MaquinaAtendimentoBO();
		   MaquinaAtendimentoVO maquinaAtendimentoVO = new
MaquinaAtendimentoVO(codMaquina);

		   // verifica se o codigo da máquina existe
		   if(maquinaAtendimentoBO.isTrue(maquinaAtendimentoVO)){
```

```
                                maquinaAtendimentoVO =
maquinaAtendimentoBO.consultar(maquinaAtendimentoVO);

                                // verifica se existe estoque do produto
                                if(maquinaAtendimentoVO.getEstoque() > 0){

                                        // verifica se cliente possui condição financeira

                                        // Valida saldo do Cliente
                                        double saldoFinal =
clienteBO.calculaSaldo(clienteVO.getSaldo(),maquinaAtendimentoVO.getValorProduto());

                                        //Em caso de cliente pós-pago verificar se nçao possui pendencia financeira
                                        if(clienteVO.getCategoria().equals("pos-pago") &&
clienteVO.getPendenciaFinanceira()!=0){
                                                sRetornoServelt = sRetornoServelt + "Cliente com Pendencia
Financeira.";
                                        //Em caso de cliente pós-pago verificar se nçao possui pendencia financeira
                                        }else if(clienteVO.getCategoria().equals("pre-pago") && saldoFinal <= 0){
                                                sRetornoServelt = sRetornoServelt + "Cliente sem saldo para
operacao.";
                                        }else{
                                                // Atualiza saldo do Cliente
                                                clienteVO = clienteBO.atualizaSaldo(clienteVO,saldoFinal);

                                                // Atualiza estoque do produto
                                                maquinaAtendimentoVO =
maquinaAtendimentoBO.atualizaEstoque(maquinaAtendimentoVO);

                                                // sucesso na operação
                                                sRetornoServelt = "Digite Cod Autorizador abaixo para liberacao do
Produto" + maquinaAtendimentoVO.getCodAutorizador();
                                        }
                                }else{
                                        sRetornoServelt = sRetornoServelt + "Produto sem estoque disponível.";
                                }
                        }else{
                                sRetornoServelt = sRetornoServelt + "Codigo de Maquina (" + codMaquina +") não
cadastrado.";
                        }
                }else{
                        sRetornoServelt = sRetornoServelt + "Falha no cadastro do Cliente.";
                }

                // Cria e envia a página para o cliente (celular)
        response.setContentType("text/vnd.wap.wml");
        PrintWriter out = response.getWriter();
        out.println("<?xml version=\"1.0\"?>");
        out.println("<!DOCTYPE wml PUBLIC " + "\"-//WAPFORUM//DTD WML 1.1//EN\" " +
"\"http://www.wapforum.org/DTD/wml_1.1.xml\">");
        out.println("<wml>");
        out.println("<card id=\"Code\">");
        out.println("<p>");
        out.println(sRetornoServelt);
        out.println("</p>");
        out.println("</card>");
        out.println("</wml>");
        out.close(); // fecha o fluxo printWriter
    }
```

```
    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the
code.">
    /** Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /** Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /** Returns a short description of the servlet.
     */
    public String getServletInfo() {
        return "Short description";
    }
    // </editor-fold>
}
```

Constantes.java

```
package projeto.operadora.util;
public class Constantes {
            public static final String CONTEXTO_JNDI =  "java:comp/env/jdbc/operadora_maquina";
            public static final String FACTORY = "org.apache.commons.dbcp.BasicDataSourceFactory";
}
```

ConnectionManager.java

```
package projeto.operadora.persistencia;

import java.sql.Connection;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.sql.DataSource;


/**
 *
 * Classe responsável pelo gerenciamento das conexões ao banco
 * @author André R. Gravina
 *
 */
public class ConnectionManager {

                private static ConnectionManager instancia = null;
```

```
	/**
	 *  Construtor private seguindo o padrão sigleton.
	 */
	private ConnectionManager(){}

	/**
	 * Método que retorna a instancia única da classe ConnectionManager
	 * @return a instancia única da classe ConnectionManager
	 */
	public static ConnectionManager getInstancia(){

		if(instancia != null){
			return instancia;
		}
		instancia = new ConnectionManager();
		return instancia;

	}

	/**
	 * Método responsável por obter uma conexão que será
	 * passada ao DAO para acesso ao banco de dados.
	 * @return a conexão para acesso ao banco
	 * @throws Exception
	 */
	public Connection abrirConexao(String JNDI) throws Exception {

		Context ctx = null;

		DataSource ds = null;
		try {
			ctx = new InitialContext();
			ds = (DataSource) ctx.lookup(JNDI);

		} catch (NamingException e) {
			throw new Exception(" Contexto não iniciado !!!!" + e.getMessage());
		}

		if(ctx == null ) {
			throw new Exception("Contexto JNDI não Iniciado !");
		}


		return ds.getConnection();
	}
	/**
	 * Devolve a conexão ao datasourceapós o uso.
	 * @param connection conexão utilizada
	 * @throws Exception
	 */
	public void fecharConexao(Connection connection) throws Exception {
		connection.close();
		connection = null;

	}
}
```

MaquinaAtendimentoBO.java

```
package projeto.operadora.negocio;

import java.sql.Connection;

import projeto.operadora.dao.MaquinaAtendimentoDAO;
import projeto.operadora.persistencia.ConnectionManager;
import projeto.operadora.util.Constantes;
import projeto.operadora.vo.MaquinaAtendimentoVO;

/**
 * MaquinaAtendimentoBO - objeto com dados do Cliente - Business Object
 *
 * Contém as regras de negócio para a máquina de auto-atendimento
 *
 * @author André R. Gravina
 *
 */

public class MaquinaAtendimentoBO {

	public MaquinaAtendimentoBO() {
	}

	public MaquinaAtendimentoVO consultar(MaquinaAtendimentoVO maquinaAtendimentoVO) {
		ConnectionManager connectionManager = null;
		Connection connection = null;
		try {
			connectionManager = ConnectionManager.getInstancia();
			connection =
ConnectionManager.getInstancia().abrirConexao(Constantes.CONTEXTO_JNDI);
			maquinaAtendimentoVO = new
MaquinaAtendimentoDAO(connection).consultar(maquinaAtendimentoVO);
			connectionManager.fecharConexao(connection);
		} catch (Exception e) {
			e.printStackTrace();
		}
		return maquinaAtendimentoVO;
	}

	public boolean isTrue(MaquinaAtendimentoVO maquinaAtendimentoVO) {
		ConnectionManager connectionManager = null;
		Connection connection = null;
		boolean retorno = false;
		try {
			connectionManager = ConnectionManager.getInstancia();
			connection =
ConnectionManager.getInstancia().abrirConexao(Constantes.CONTEXTO_JNDI);
			retorno = new
MaquinaAtendimentoDAO(connection).isTrue(maquinaAtendimentoVO);
			connectionManager.fecharConexao(connection);
		} catch (Exception e) {
			e.printStackTrace();
		}
		return retorno;
	}

	public MaquinaAtendimentoVO atualizaEstoque(MaquinaAtendimentoVO maquinaAtendimentoVO) {
			ConnectionManager connectionManager = null;
```

```
                        Connection connection = null;
                        int estoque = maquinaAtendimentoVO.getEstoque();
                        estoque = estoque - 1;
                        maquinaAtendimentoVO.setEstoque(estoque);
                        try {
                                connectionManager = ConnectionManager.getInstancia();
                                connection =
ConnectionManager.getInstancia().abrirConexao(Constantes.CONTEXTO_JNDI);
                                maquinaAtendimentoVO = new
MaquinaAtendimentoDAO(connection).atualizaEstoque(maquinaAtendimentoVO);
                                connectionManager.fecharConexao(connection);
                        } catch (Exception e) {
                                e.printStackTrace();
                        }
                return maquinaAtendimentoVO;
        }

}
```

ClienteBO.java

```
package projeto.operadora.negocio;

import java.sql.Connection;

import projeto.operadora.dao.ClienteDAO;
import projeto.operadora.persistencia.ConnectionManager;
import projeto.operadora.util.Constantes;
import projeto.operadora.vo.ClienteVO;

/**
 * ClienteBO - objeto com dados do Cliente - Business Object
 *
 * @author André R. Gravina
 *
 */
public class ClienteBO {

        public ClienteBO() {
        }

        public void excluir(Object objeto) throws Exception {
                ConnectionManager connectionManager = null;
                Connection connection = null;
                try {
                        connectionManager = ConnectionManager.getInstancia();
                        connection =
ConnectionManager.getInstancia().abrirConexao(Constantes.CONTEXTO_JNDI);
                        connection.setAutoCommit(false);
                        connection.setAutoCommit(true);
                        connectionManager.fecharConexao(connection);
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }

        public ClienteVO consultar(ClienteVO clienteVO) {
```

```
		ConnectionManager connectionManager = null;
		Connection connection = null;
		try {
			connectionManager = ConnectionManager.getInstancia();
			connection =
ConnectionManager.getInstancia().abrirConexao(Constantes.CONTEXTO_JNDI);
			clienteVO = new ClienteDAO(connection).consultar(clienteVO);
			connectionManager.fecharConexao(connection);
		} catch (Exception e) {
			e.printStackTrace();
		}
		return clienteVO;
	}

	public ClienteVO atualizaSaldo(ClienteVO clienteVO, double saldo) {
		ConnectionManager connectionManager = null;
		Connection connection = null;
		clienteVO.setSaldo(saldo);
		try {
			connectionManager = ConnectionManager.getInstancia();
			connection =
ConnectionManager.getInstancia().abrirConexao(Constantes.CONTEXTO_JNDI);
			clienteVO = new ClienteDAO(connection).atualizaSaldo(clienteVO);
			connectionManager.fecharConexao(connection);
		} catch (Exception e) {
			e.printStackTrace();
		}
		return clienteVO;
	}

	public boolean isTrue(ClienteVO clienteVO) {
		ConnectionManager connectionManager = null;
		Connection connection = null;
		boolean retorno = false;
		try {
			connectionManager = ConnectionManager.getInstancia();
			connection =
ConnectionManager.getInstancia().abrirConexao(Constantes.CONTEXTO_JNDI);
			retorno = new ClienteDAO(connection).isTrue(clienteVO);
			connectionManager.fecharConexao(connection);
		} catch (Exception e) {
			e.printStackTrace();
		}
		return retorno;
	}

	public double calculaSaldo(double saldo, double valorProduto) {
		return (saldo-valorProduto);
	}

}
```

ClienteDAO.java

```
package projeto.operadora.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import projeto.operadora.vo.ClienteVO;

/**
 * ServidorOperadoraDAO Data Access Object
 *
 * @author André R. Gravina
 *
 */

public class ClienteDAO {
	private Connection conexao = null;

	public ClienteDAO(Connection connection) {
		this.conexao = connection;
	}

	public boolean isTrue(ClienteVO clienteVO) {
		StringBuffer sql = new StringBuffer();
		PreparedStatement preparedStatement = null;
		ResultSet resultado = null;
		boolean retorno = false;

		sql.append("SELECT * FROM clientes where msisdn = ?");
		try {
			preparedStatement = this.conexao.prepareStatement(sql.toString());
			preparedStatement.setString(1, clienteVO.getMsisdn());
			try {
				resultado = preparedStatement.executeQuery();
			} catch (SQLException e1) {
				e1.printStackTrace();
			}

			while (resultado.next()) {
				retorno = true;
			}

		} catch (SQLException e) {
			e.printStackTrace();
		} finally {
			try {
				if (resultado != null)
					resultado.close();
			} catch (SQLException e) {
				e.printStackTrace();
			}
			try {
				if (preparedStatement != null)
					preparedStatement.close();
			} catch (SQLException e) {
				e.printStackTrace();
			}
```

```
		}

		return retorno;

	}
	public ClienteVO atualizaSaldo(ClienteVO clienteVO) {

		PreparedStatement preparedStatement = null;

		try {
			String sqlUpdate = "UPDATE clientes set saldo = ? WHERE msisdn = ?";
			preparedStatement = this.conexao.prepareStatement(sqlUpdate);
			preparedStatement.setDouble(1, clienteVO.getSaldo());
			preparedStatement.setString(2, clienteVO.getMsisdn());
			preparedStatement.executeUpdate();

		} catch (SQLException e) {
			e.printStackTrace();
		} finally {
			try {
				if (preparedStatement != null)
					preparedStatement.close();
			} catch (SQLException e) {
				e.printStackTrace();
			}
		}
		return clienteVO;
	}


	protected ClienteVO fillVo(ResultSet rs) {
		ClienteVO fillClienteVO = new ClienteVO();
		try {
			fillClienteVO.setCategoria(rs.getString("categoria"));
			fillClienteVO.setEndereco(rs.getString("endereco"));
			fillClienteVO.setMsisdn(rs.getString("msisdn"));
			fillClienteVO.setNome(rs.getString("nome"));
			fillClienteVO.setPendenciaFinanceira(rs.getInt("pendencia_financeira"));
			fillClienteVO.setSaldo(rs.getDouble("saldo"));
		} catch (SQLException e) {
			e.printStackTrace();
		}
		return fillClienteVO;
	}

	public ClienteVO consultar(ClienteVO clienteVO) {
		StringBuffer sql = new StringBuffer();
		PreparedStatement preparedStatement = null;
		ResultSet resultado = null;

		sql.append("SELECT * FROM clientes where msisdn = ?");
		try {
			preparedStatement = this.conexao.prepareStatement(sql.toString());
			preparedStatement.setString(1, clienteVO.getMsisdn());
			try {
				resultado = preparedStatement.executeQuery();
			} catch (SQLException e1) {
				e1.printStackTrace();
			}
```

```
                    while (resultado.next()) {
                        clienteVO = fillVo(resultado);
                    }

                } catch (SQLException e) {
                    e.printStackTrace();
                } finally {
                    try {
                        if (resultado != null)
                            resultado.close();
                    } catch (SQLException e) {
                        e.printStackTrace();
                    }
                    try {
                        if (preparedStatement != null)
                            preparedStatement.close();
                    } catch (SQLException e) {
                        e.printStackTrace();
                    }
                }
                return clienteVO;
        }

}
```

MaquinaAtendimentoDAO.java

```
package projeto.operadora.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import projeto.operadora.vo.MaquinaAtendimentoVO;

/**
 * MaquinaAtendimentoDAO Data Access Object
 * Acesso ao Banco para retornar dados da Máquina de Auto - Atendimento
 *
 * @author André R. Gravina
 *
 */

public class MaquinaAtendimentoDAO {
        private Connection conexao = null;

        public MaquinaAtendimentoDAO(Connection connection) {
                this.conexao = connection;
        }

        public boolean isTrue(MaquinaAtendimentoVO maquinaAtendimentoVO) {
                StringBuffer sql = new StringBuffer();
                PreparedStatement preparedStatement = null;
                ResultSet resultado = null;
                boolean retorno = false;

                sql.append("SELECT 1 FROM auto_atendimento where cod_maquina = ?");
                try {
```

```
                preparedStatement = this.conexao.prepareStatement(sql.toString());
                preparedStatement.setString(1, maquinaAtendimentoVO.getCodMaquina());
                try {
                        resultado = preparedStatement.executeQuery();
                } catch (SQLException e1) {
                        e1.printStackTrace();
                }

                while (resultado.next()) {
                        retorno = true;
                }

        } catch (SQLException e) {
                e.printStackTrace();
        } finally {
                try {
                        if (resultado != null)
                                resultado.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                }
                try {
                        if (preparedStatement != null)
                                preparedStatement.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }

        return retorno;

}

protected MaquinaAtendimentoVO fillVo(ResultSet rs) {
        MaquinaAtendimentoVO fillMaquinaAtendimentoVO = new MaquinaAtendimentoVO();
        try {
                fillMaquinaAtendimentoVO.setCodAutorizador(rs.getString("cod_autorizador"));
                fillMaquinaAtendimentoVO.setProduto(rs.getString("produto"));

                fillMaquinaAtendimentoVO.setTipoProduto(rs.getString("tipo_produto"));
                fillMaquinaAtendimentoVO.setValorProduto(rs.getDouble("valor_produto"));
                fillMaquinaAtendimentoVO.setCodMaquina(rs.getString("cod_maquina"));
                fillMaquinaAtendimentoVO.setEstoque(rs.getInt("estoque"));
        } catch (SQLException e) {
                e.printStackTrace();
        }
        return fillMaquinaAtendimentoVO;
}

public MaquinaAtendimentoVO consultar(MaquinaAtendimentoVO maquinaAtendimentoVO) {
        StringBuffer sql = new StringBuffer();
        PreparedStatement preparedStatement = null;
        ResultSet resultado = null;

        sql.append("SELECT * FROM auto_atendimento where cod_maquina = ?");
        //sql.append("SELECT * FROM auto_atendimento");
        try {
                preparedStatement = this.conexao.prepareStatement(sql.toString());
                preparedStatement.setString(1, maquinaAtendimentoVO.getCodMaquina());
                try {
```

```
                        resultado = preparedStatement.executeQuery();
                    } catch (SQLException e1) {
                        e1.printStackTrace();
                    }

                    while (resultado.next()) {
                        maquinaAtendimentoVO = fillVo(resultado);
                        //System.out.println(resultado.getString(1));
                    }

                } catch (SQLException e) {
                    e.printStackTrace();
                } finally {
                    try {
                        if (resultado != null)
                            resultado.close();
                    } catch (SQLException e) {
                        e.printStackTrace();
                    }
                    try {
                        if (preparedStatement != null)
                            preparedStatement.close();
                    } catch (SQLException e) {
                        e.printStackTrace();
                    }
                }
                return maquinaAtendimentoVO;
        }

        public MaquinaAtendimentoVO atualizaEstoque(MaquinaAtendimentoVO maquinaAtendimentoVO) {
                    PreparedStatement preparedStatement = null;

                    try {
                        String sqlUpdate = "UPDATE auto_atendimento set estoque = ? WHERE
cod_maquina = ?";
                        preparedStatement = this.conexao.prepareStatement(sqlUpdate);
                        preparedStatement.setDouble(1, maquinaAtendimentoVO.getEstoque());
                        preparedStatement.setString(2, maquinaAtendimentoVO.getCodMaquina());
                        preparedStatement.executeUpdate();

                    } catch (SQLException e) {
                        e.printStackTrace();
                    } finally {
                        try {
                            if (preparedStatement != null)
                                preparedStatement.close();
                        } catch (SQLException e) {
                            e.printStackTrace();
                        }
                    }
                    return maquinaAtendimentoVO;
        }

}
```

MaquinaAtendimentoVO.java

```
package projeto.operadora.vo;

import java.io.Serializable;

public class MaquinaAtendimentoVO implements Serializable {
	private static final long serialVersionUID = 1L;
	private String codMaquina;
	private String tipoProduto;
	private String produto;
	private String codAutorizador;
	private double valorProduto;
	private int estoque;


	/**
	 * @return Returns the estoque.
	 */
	public int getEstoque() {
		return estoque;
	}
	/**
	 * @param estoque The estoque to set.
	 */
	public void setEstoque(int estoque) {
		this.estoque = estoque;
	}
	public MaquinaAtendimentoVO(){}
	public MaquinaAtendimentoVO(String codMaquina){
		this.codMaquina = codMaquina;
	}
	/**
	 * @return Returns the codAutorizador.
	 */
	public String getCodAutorizador() {
		return codAutorizador;
	}
	/**
	 * @param codAutorizador The codAutorizador to set.
	 */
	public void setCodAutorizador(String codAutorizador) {
		this.codAutorizador = codAutorizador;
	}

	/**
	 * @return Returns the codMaquina.
	 */
	public String getCodMaquina() {
		return codMaquina;
	}
	/**
	 * @param codMaquina The codMaquina to set.
	 */
	public void setCodMaquina(String codMaquina) {
		this.codMaquina = codMaquina;
	}
	/**
	 * @return Returns the marca.
	 */
```

```
	public String getProduto() {
		return produto;
	}
	/**
	 * @param marca The marca to set.
	 */
	public void setProduto(String produto) {
		this.produto = produto;
	}
	/**
	 * @return Returns the tipoProduto.
	 */
	public String getTipoProduto() {
		return tipoProduto;
	}
	/**
	 * @param tipoProduto The tipoProduto to set.
	 */
	public void setTipoProduto(String tipoProduto) {
		this.tipoProduto = tipoProduto;
	}
	/**
	 * @return Returns the valorProduto.
	 */
	public double getValorProduto() {
		return valorProduto;
	}
	/**
	 * @param valorProduto The valorProduto to set.
	 */
	public void setValorProduto(double valorProduto) {
		this.valorProduto = valorProduto;
	}
}
```

ClienteVO.java

```
package projeto.operadora.vo;

import java.io.Serializable;

public class ClienteVO implements Serializable {
	private static final long serialVersionUID = 1L;
	private String nome;
	private String endereco;
	private String msisdn;
	private String categoria;
	private double saldo;
	private int pendenciaFinanceira;

	public ClienteVO(){}
	public ClienteVO(String msisdn){
		this.msisdn = msisdn;
	}

	/**
	 * @return Returns the categoria.
	 */
```

```
public String getCategoria() {
	return categoria;
}
/**
 * @param categoria The categoria to set.
 */
public void setCategoria(String categoria) {
	this.categoria = categoria;
}
/**
 * @return Returns the endereco.
 */
public String getEndereco() {
	return endereco;
}
/**
 * @param endereco The endereco to set.
 */
public void setEndereco(String endereco) {
	this.endereco = endereco;
}
/**
 * @return Returns the msisdn.
 */
public String getMsisdn() {
	return msisdn;
}
/**
 * @param msisdn The msisdn to set.
 */
public void setMsisdn(String msisdn) {
	this.msisdn = msisdn;
}
/**
 * @return Returns the nome.
 */
public String getNome() {
	return nome;
}
/**
 * @param nome The nome to set.
 */
public void setNome(String nome) {
	this.nome = nome;
}
/**
 * @return Returns the pendenciaFinanceira.
 */
public int getPendenciaFinanceira() {
	return pendenciaFinanceira;
}
/**
 * @param pendenciaFinanceira The pendenciaFinanceira to set.
 */
public void setPendenciaFinanceira(int pendenciaFinanceira) {
	this.pendenciaFinanceira = pendenciaFinanceira;
}
/**
 * @return Returns the saldo.
 */
```

```
	public double getSaldo() {
		return saldo;
	}
	/**
	 * @param saldo The saldo to set.
	 */
	public void setSaldo(double saldo) {
		this.saldo = saldo;
	}


}
```

Anexo B. CÓDIGO FONTE DO SIMULADOR DA MÁQUINA DE AUTO-ATENDIMENTO

AutoAtendimento.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.JTextArea;
import projeto.serialcom.ComunicaPSerial;
import projeto.serialcom.PortaSerial;
import projeto.util.Constantes;

/**
 * AutoAtendimento.java
 * Simulador da máquina de Auto Atendimento
 * @author: André R . Gravina
 * 20/06/2006
 */

public class AutoAtendimento extends javax.swing.JFrame {

    //Declaração de constantes
    public final String FAVOR_SELECIONAR = "Favor Selecionar";
    public final String REFRI_COLA = "Refrigerante Cola";
    public final String REFRI_GUARANA = "Refrigerante Guarana";
    public final String CAFE_LONGO = "Cafe Longo";
    public final String CAFE_EXPRESSO = "Cafe Expresso";
    public final String COD_AUTORIZADOR_REFRI_COLA = "1000000001";
    public final String COD_AUTORIZADOR_REFRI_GUARANA = "1000000002";
    public final String COD_AUTORIZADOR_CAFE_LONGO = "1000000003";
    public final String COD_AUTORIZADOR_CAFE_EXPRESSO = "1000000004";
    public final String PORTA_SERIAL = "COM1";
    public final int PORTA_VELOCIDADE = 9600;
    public final int PORTA_TIME_OUT = 2000;

    // Variáveis Modulares

    // Código digitado pelo usuário para liberação do produto
    private String codigoAutorizacao = "" ;
    // Código que deve ser digitado pelo usuário para liberação do produto
    private String codigoRequerido = "";
    private int tipoProduto = 0;
    private String produto = "";

    private class ButtonHandler implements ActionListener {

          public void actionPerformed(ActionEvent e) {

        if(e.getActionCommand().equals("Corrigir")){
           if(codigoAutorizacao.length() > 0){
              String txtAuxiliar = "";
              txtAuxiliar = "Digite o Codigo Autorizador:\n";
              jTextArea1.setText(txtAuxiliar);
              codigoAutorizacao = "";
           }
        }else{
           if(codigoAutorizacao.length() < 10){
```

```
            jTextArea1.append(e.getActionCommand());
            codigoAutorizacao+=e.getActionCommand();
          }
      }

    }

}

private class KeyHandler
    implements KeyListener
{

    public void keyPressed(KeyEvent e)
    {
      if(codigoAutorizacao.length() < 10){
        String s = "";
          s+= jTextArea1.getText() + e.getKeyChar();
        jTextArea1.setText(s);
        codigoAutorizacao+=e.getKeyChar();
      }
    }

    public void keyReleased(KeyEvent e)
    {
    }

    public void keyTyped(KeyEvent keyevent)
    {
    }

    KeyHandler()
    {
    }
}
private class ChoiceTipoCriptoHandler implements ItemListener{

    public void itemStateChanged(ItemEvent e){
        codigoRequerido = "";
        tipoProduto = jCmbProdutos.getSelectedIndex();
        produto = Integer.toString(tipoProduto);

        if(tipoProduto > 0){
            jTextArea1.setText("Digite o Codigo Autorizador:\n");
        }

        switch(tipoProduto){

            // REFRI_COLA
            case 1:
                txtCodMaquina.setText("000001");
                codigoRequerido = COD_AUTORIZADOR_REFRI_COLA;
                break;
            // REFRI_GUARANA
            case 2:
                txtCodMaquina.setText("000002");
                codigoRequerido = COD_AUTORIZADOR_REFRI_GUARANA;
                break;
            // CAFE_LONGO
            case 3:
```

```
                txtCodMaquina.setText("000003");
                codigoRequerido = COD_AUTORIZADOR_CAFE_LONGO;
                break;
            // CAFE_EXPRESSO
            case 4:
                txtCodMaquina.setText("000004");
                codigoRequerido = COD_AUTORIZADOR_CAFE_EXPRESSO;
                break;
            // ERRO
            default:
                txtCodMaquina.setText("------");
                produto = "0";
                break;
        }
    }
}
/** Creates new form AutoAtendimento */
public AutoAtendimento() {
    initComponents();

    jBtn1.addActionListener(new ButtonHandler());
    jBtn2.addActionListener(new ButtonHandler());
    jBtn3.addActionListener(new ButtonHandler());
    jBtn4.addActionListener(new ButtonHandler());
    jBtn5.addActionListener(new ButtonHandler());
    jBtn6.addActionListener(new ButtonHandler());
    jBtn7.addActionListener(new ButtonHandler());
    jBtn8.addActionListener(new ButtonHandler());
    jBtn9.addActionListener(new ButtonHandler());
    jBtn0.addActionListener(new ButtonHandler());
    jBtnCorrigir.addActionListener(new ButtonHandler());
        jBtn1.addKeyListener(new KeyHandler());
    jBtn2.addKeyListener(new KeyHandler());
    jBtn3.addKeyListener(new KeyHandler());
    jBtn4.addKeyListener(new KeyHandler());
    jBtn5.addKeyListener(new KeyHandler());
    jBtn6.addKeyListener(new KeyHandler());
    jBtn7.addKeyListener(new KeyHandler());
    jBtn8.addKeyListener(new KeyHandler());
    jBtn9.addKeyListener(new KeyHandler());
    jBtn0.addKeyListener(new KeyHandler());
    jBtnCorrigir.addKeyListener(new KeyHandler());
    jTextArea1.setText("Selecione um Produto.\n");
    jCmbProdutos.addItemListener(new ChoiceTipoCriptoHandler());



}

/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
// <editor-fold defaultstate="collapsed" desc=" Generated Code ">
private void initComponents() {
    jCmbProdutos = new javax.swing.JComboBox();
    jButtonLibProduto = new javax.swing.JButton();
    jPanelTeclado = new javax.swing.JPanel();
    jBtn1 = new javax.swing.JButton();
```

```
        jBtn2 = new javax.swing.JButton();
        jBtn3 = new javax.swing.JButton();
        jBtn4 = new javax.swing.JButton();
        jBtn5 = new javax.swing.JButton();
        jBtn6 = new javax.swing.JButton();
        jBtn7 = new javax.swing.JButton();
        jBtn8 = new javax.swing.JButton();
        jBtn9 = new javax.swing.JButton();
        jBtn0 = new javax.swing.JButton();
        jBtnCorrigir = new javax.swing.JButton();
        jLabel1 = new javax.swing.JLabel();
        jPanel1 = new javax.swing.JPanel();
        txtCodMaquina = new javax.swing.JTextField();
        lblCodMaquina = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        jMenuBar1 = new javax.swing.JMenuBar();
        jMenu1 = new javax.swing.JMenu();
        Sair = new javax.swing.JMenuItem();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Simula\u00e7\u00e3o - M\u00e1quina de Auto-Atendimento");
        setResizable(false);
        jCmbProdutos.setFont(new java.awt.Font("Arial", 0, 12));
        jCmbProdutos.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "Favor Selecionar",
"Refrigerante Cola", "Refrigerante Guaran\u00e1", "Caf\u00e9 Longo", "Caf\u00e9 Expresso" }));
        jCmbProdutos.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jCmbProdutosActionPerformed(evt);
            }
        });

        jButtonLibProduto.setFont(new java.awt.Font("Tahoma", 1, 14));
        jButtonLibProduto.setText("Liberar Produto");
        jButtonLibProduto.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButtonLibProdutoActionPerformed(evt);
            }
        });

        jPanelTeclado.setBorder(javax.swing.BorderFactory.createEtchedBorder());
        jBtn1.setText("1");
        jBtn1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jBtn1ActionPerformed(evt);
            }
        });

        jBtn2.setText("2");
        jBtn2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jBtn2ActionPerformed(evt);
            }
        });

        jBtn3.setText("3");

        jBtn4.setText("4");

        jBtn5.setText("5");
```

```
        jBtn6.setText("6");

        jBtn7.setText("7");

        jBtn8.setText("8");
        jBtn8.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jBtn8ActionPerformed(evt);
            }
        });

        jBtn9.setText("9");

        jBtn0.setText("0");

        jBtnCorrigir.setFont(new java.awt.Font("Tahoma", 1, 10));
        jBtnCorrigir.setText("Corrigir");

        org.jdesktop.layout.GroupLayout jPanelTecladoLayout = new
org.jdesktop.layout.GroupLayout(jPanelTeclado);
        jPanelTeclado.setLayout(jPanelTecladoLayout);
        jPanelTecladoLayout.setHorizontalGroup(
            jPanelTecladoLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jPanelTecladoLayout.createSequentialGroup()
                .addContainerGap()
                .add(jPanelTecladoLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                    .add(jPanelTecladoLayout.createSequentialGroup()
                        .add(jBtn4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 71,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(jBtn5, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 71,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(jPanelTecladoLayout.createSequentialGroup()
                        .add(jBtn7, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 71,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(jBtn8, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 71,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(jPanelTecladoLayout.createSequentialGroup()
                        .add(jBtn1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 71,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(jBtn2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 71,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(org.jdesktop.layout.GroupLayout.TRAILING, jPanelTecladoLayout.createSequentialGroup()
                        .add(jBtn0, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 148,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)))
                .add(14, 14, 14)
                .add(jPanelTecladoLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING, false)
                    .add(jBtn3, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 71, Short.MAX_VALUE)
                    .add(jBtn6, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 71, Short.MAX_VALUE)
                    .add(jBtn9, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 71, Short.MAX_VALUE)
                    .add(jBtnCorrigir, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap(18, Short.MAX_VALUE))
        );
        jPanelTecladoLayout.setVerticalGroup(
            jPanelTecladoLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
```

```
            .add(org.jdesktop.layout.GroupLayout.TRAILING, jPanelTecladoLayout.createSequentialGroup()
                .addContainerGap(14, Short.MAX_VALUE)
                .add(jPanelTecladoLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jBtn1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(jBtn2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(jBtn3, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jPanelTecladoLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jBtn4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(jBtn5, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(jBtn6, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jPanelTecladoLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jBtn7, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(jBtn9, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(jBtn8, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jPanelTecladoLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jBtn0, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(jBtnCorrigir, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 35,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );

        jLabel1.setFont(new java.awt.Font("Tahoma", 0, 12));
        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setText("Escolha o Produto");

        jPanel1.setBorder(javax.swing.BorderFactory.createEtchedBorder());
        txtCodMaquina.setEditable(false);
        txtCodMaquina.setFont(new java.awt.Font("Tahoma", 1, 14));
        txtCodMaquina.setHorizontalAlignment(javax.swing.JTextField.CENTER);
        txtCodMaquina.setText("------");

        lblCodMaquina.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        lblCodMaquina.setText("C\u00f3digo da M\u00e1quina");

        org.jdesktop.layout.GroupLayout jPanel1Layout = new org.jdesktop.layout.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(lblCodMaquina, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 151, Short.MAX_VALUE)
            .add(org.jdesktop.layout.GroupLayout.TRAILING, jPanel1Layout.createSequentialGroup()
                .addContainerGap(49, Short.MAX_VALUE)
                .add(txtCodMaquina, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 68,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .add(34, 34, 34))
        );
        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
```

```
            .add(jPanel1Layout.createSequentialGroup()
                .add(lblCodMaquina)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(txtCodMaquina, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        jTextArea1.setColumns(20);
        jTextArea1.setEditable(false);
        jTextArea1.setLineWrap(true);
        jTextArea1.setRows(5);
        jScrollPane1.setViewportView(jTextArea1);

        jMenu1.setText("Menu");
        Sair.setText("Sair");
        Sair.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                SairActionPerformed(evt);
            }
        });

        jMenu1.add(Sair);

        jMenuBar1.add(jMenu1);

        setJMenuBar(jMenuBar1);

        org.jdesktop.layout.GroupLayout layout = new org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                    .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 267, Short.MAX_VALUE)
                    .add(jPanelTeclado, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING, false)
                    .add(jCmbProdutos, 0, 155, Short.MAX_VALUE)
                    .add(jLabel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .add(jButtonLibProduto, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                    .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 57, Short.MAX_VALUE)
                    .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                    .add(layout.createSequentialGroup()
```

```
                    .add(jLabel1)
                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                    .add(jCmbProdutos, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                    .add(jButtonLibProduto))
                .add(jPanelTeclado, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
            .add(25, 25, 25))
        );
        pack();
    }// </editor-fold>

    private void jButtonLibProdutoActionPerformed(java.awt.event.ActionEvent evt) {

        if(tipoProduto <= 0 ){
            jTextArea1.setText("Selecione um Produto.");
        }
        //Valida Código Autorizador
        else if(codigoAutorizacao.length() > 0 && codigoAutorizacao.equals(codigoRequerido)){

            // realiza tratamento com a porta serial do computador
            PortaSerial ps = new PortaSerial();
            if ( ps.PortaExiste(PORTA_SERIAL) == true) {
                System.out.println("Iniciando comunicação!");
                ComunicaPSerial sc = new
ComunicaPSerial(PORTA_SERIAL,PORTA_VELOCIDADE,PORTA_TIME_OUT);
                sc.HabilitarEscrita();
                sc.ObterIdDaPorta();
                sc.AbrirPorta();
                sc.EnviarUmaString(produto);
                sc.FecharCom();
            }
        }else{
            // Exibe mensagem de erro e limpa veriáveis de controle
            jTextArea1.setText("CÓDIGO INVÁLIDO!\n");
            jTextArea1.append("Digite o Codigo Autorizador:\n");
            codigoAutorizacao = "";
        }
    }

    private void jBtn2ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
    }

    private void jCmbProdutosActionPerformed(java.awt.event.ActionEvent evt) {
    // desabitiar botões

    }

    private void jBtn1ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
    }

    private void jBtn8ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
    }

    private void SairActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
```

```
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new AutoAtendimento().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JMenuItem Sair;
    private javax.swing.JButton jBtn0;
    private javax.swing.JButton jBtn1;
    private javax.swing.JButton jBtn2;
    private javax.swing.JButton jBtn3;
    private javax.swing.JButton jBtn4;
    private javax.swing.JButton jBtn5;
    private javax.swing.JButton jBtn6;
    private javax.swing.JButton jBtn7;
    private javax.swing.JButton jBtn8;
    private javax.swing.JButton jBtn9;
    private javax.swing.JButton jBtnCorrigir;
    private javax.swing.JButton jButtonLibProduto;
    private javax.swing.JComboBox jCmbProdutos;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JMenu jMenu1;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanelTeclado;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextArea jTextArea1;
    private javax.swing.JLabel lblCodMaquina;
    private javax.swing.JTextField txtCodMaquina;
    // End of variables declaration
    }
```

ComunicaPSerial.java

```
package projeto.serialcom;

import javax.comm.*;
import java.io.*;

/**
 ********************************************************************************
 * REALIZI COMUNICAÇÃO COM A PORTA SERIAL / Java communications API
 * Classe  PortaSerial.java
 * @autor André R. Gravina
 * @data 01/07/2006
  ********************************************************************************
 */

//classe Principal
public class ComunicaPSerial implements Runnable, SerialPortEventListener {
    //propriedades
```

```
public String Dadoslidos;
public int nodeBytes;
private String Porta;
private int baudrate;
private int timeout;
private CommPortIdentifier cp;
private SerialPort porta;
private OutputStream saida;
private InputStream entrada;
private Thread threadLeitura;

//indicadores
private boolean IDPortaOK; //true porta EXISTE
private boolean PortaOK;// true porta aberta
private boolean Leitura;
private boolean Escrita;

//construtor default paridade : par
//baudrate: 9600 bps stopbits: 2 COM 1
public ComunicaPSerial() {
   Porta = "COM1";
   baudrate = 9600;
   timeout = 1000;
};

//um Objeto ComObj é passado ao construtor
//com detalhes de qual porta abrir
//e informações sobre configurações
public ComunicaPSerial( String p , int b , int t ){
   this.Porta = p;
   this.baudrate = b;
   this.timeout = t;
};

//habilita escrita de dados
public void HabilitarEscrita(){
   Escrita = true;
   Leitura = false;
}

//habilita leitura de dados
public void HabilitarLeitura(){
   Escrita = false;
   Leitura = true;
}
//Obtém o ID da PORTA
public void ObterIdDaPorta(){
   try {
      cp = CommPortIdentifier.getPortIdentifier(Porta);
      if ( cp == null ) {
         System.out.println("A " + Porta + " nao existe!" );
         IDPortaOK = false;
         System.exit(1);
      }
      IDPortaOK = true;
   } catch (Exception e) {
      System.out.println("Erro durante o procedimento. STATUS" + e );
      IDPortaOK = false;
      System.exit(1);
```

```
    }
}

//Abre a comunicação da porta
public void AbrirPorta(){
    try {
        porta = (SerialPort)cp.open("SComm",timeout);
        PortaOK = true;
        System.out.println("Porta aberta com sucesso!");

        //configurar parâmetros
        porta.setSerialPortParams(baudrate,
        porta.DATABITS_8,
        porta.STOPBITS_2,
        porta.PARITY_NONE);
    } catch (Exception e) {
        PortaOK = false;
        System.out.println("Erro ao abrir a porta! STATUS: " + e );
        System.exit(1);
    }
}

//função que envie um bit para a porta serial
public void EnviarUmaString(String msg){
    if (Escrita==true) {
        try {
            saida = porta.getOutputStream();
            System.out.println("FLUXO OK!");
        } catch (Exception e) {
            System.out.println("Erro.STATUS: " + e );
        }
        try {
            System.out.println("Enviando um byte para " + Porta );
            System.out.println("Enviando : " + msg );
            saida.write(msg.getBytes());
            Thread.sleep(100);
            saida.flush();
        } catch (Exception e) {
            System.out.println("Houve um erro durante o envio. ");
            System.out.println("STATUS: " + e );
            System.exit(1);
        }
    } else {
        System.exit(1);
        }
}

//leitura de dados na serial
public void LerDados(){
    if (Escrita == true){
        try {
            entrada = porta.getInputStream();
            System.out.println("FLUXO OK!");
        } catch (Exception e) {
            System.out.println("Erro.STATUS: " + e );
            System.exit(1);
        }
        try {
            porta.addEventListener(this);
            System.out.println("SUCESSO. Porta aguardando...");
```

```
            } catch (Exception e) {
                System.out.println("Erro ao criar listener: ");
                System.out.println("STATUS: " + e);
                System.exit(1);
            }
            porta.notifyOnDataAvailable(true);
            try {
                threadLeitura = new Thread(this);
                threadLeitura.start();
            } catch (Exception e) {
                System.out.println("Erro ao iniciar leitura: " + e );
            }
        }
    }

    //método RUN da thread de leitura
    public void run(){
        try {
            Thread.sleep(5000);
        } catch (Exception e) {
            System.out.println("Erro. Status = " + e );
    }
}

    //gerenciador de eventos de leitura na serial
    public void serialEvent(SerialPortEvent ev){
        switch (ev.getEventType()) {
            case SerialPortEvent.BI:
            case SerialPortEvent.OE:
            case SerialPortEvent.FE:
            case SerialPortEvent.PE:
            case SerialPortEvent.CD:
            case SerialPortEvent.CTS:
            case SerialPortEvent.DSR:
            case SerialPortEvent.RI:
            case SerialPortEvent.OUTPUT_BUFFER_EMPTY:
                break;
            case SerialPortEvent.DATA_AVAILABLE:
                byte[] bufferLeitura = new byte[20];
                try {
                    while ( entrada.available() > 0 ) {
                        nodeBytes = entrada.read(bufferLeitura);
                    }
                    String Dadoslidos = new String(bufferLeitura);
                    if (bufferLeitura.length == 0) {
                        System.out.println("Nada lido!");
                    } else if (bufferLeitura.length == 1 ){
                        System.out.println("Apenas um byte foi lido!");
                    } else {
                        System.out.println(Dadoslidos);
                    }
                } catch (Exception e) {
                    System.out.println("Erro durante a leitura: " + e );
                }
                System.out.println("n.o de bytes lidos : " + nodeBytes );
                break;
        }
    }

    //função que fecha a conexão
```

```
    public void FecharCom(){
        try {
            porta.close();
            System.out.println("CONEXAO FECHADA>>FIM..");
        } catch (Exception e) {
            System.out.println("ERRO AO FECHAR. STATUS: " + e );
            System.exit(0);
        }
    }

    //Acessores
    public String obterPorta(){
        return Porta;
    }

    public int obterBaudrate(){
        return baudrate;
    }
}
```

PortaSerial.java

```
package projeto.serialcom;

import javax.comm.*;
import java.util.*;

/**
 ********************************************************************************
 * VERIFICA PORTAS PARA COMUNICAÇÃO SERIAL / Java communications API
 * Classe  PortaSerial.java
 * @autor André R. Gravina
 * @data 01/07/2006
 ********************************************************************************
 */

public class PortaSerial {

    //variáveis para identificar portas
    protected String[] portas;
    protected Enumeration listaDePortas;

    //Construtor
    public PortaSerial(){
        listaDePortas = CommPortIdentifier.getPortIdentifiers();
    }

    //Retorna as portas disponíveis
    public String[] ObterPortas(){
        return portas;
    }

    //Copia portas disponíveispara um Array
    private void ListarPortas(){
        int i = 0;
        portas = new String[10];
        while (listaDePortas.hasMoreElements()) {
            CommPortIdentifier ips =
            (CommPortIdentifier)listaDePortas.nextElement();
```

```
            portas[i] = ips.getName();
            i++;
        }
    }

    //Verifica se a Porta existe
    public boolean PortaExiste(String COMp){
        String temp;
        boolean e = false;
        while (listaDePortas.hasMoreElements()) {
            CommPortIdentifier ips =
            (CommPortIdentifier)listaDePortas.nextElement();
            temp = ips.getName();
            if (temp.equals(COMp)== true) {
                e = true;
            }
        }
        return e;
    }
}//FIM DA CLASSE
```

Anexo C. SCRIPT DE CRIAÇÃO DO BANCO DE DADOS

```
    -- MySQL Administrator dump 1.4
--
-- ------------------------------------------------------
-- Server version  5.0.18-nt


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;

/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;


--
-- Create schema operadora_maquina
--

CREATE DATABASE /*!32312 IF NOT EXISTS*/ operadora_maquina;
USE operadora_maquina;

--
-- Table structure for table `operadora_maquina`.`auto_atendimento`
--

DROP TABLE IF EXISTS `auto_atendimento`;
CREATE TABLE `auto_atendimento` (
  `cod_maquina` varchar(6) NOT NULL default '' COMMENT 'Identifica a máquina de autoatendimento e o
produto comercializado',
  `produto` varchar(45) NOT NULL default '' COMMENT 'Tipo de Produto',
  `tipo_produto` varchar(45) NOT NULL default '' COMMENT 'marca do produto',
```

```
  `cod_autorizador` varchar(10) NOT NULL default '' COMMENT 'Contem o codigo autorizador da compra',
  `valor_produto` decimal(8,2) NOT NULL default '0.00' COMMENT 'preço do produto',
  `estoque` decimal(10,0) NOT NULL default '0' COMMENT 'estoque do produto',
  PRIMARY KEY  (`cod_maquina`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 COMMENT='Controle das maquinas de auto atendimento';

--
-- Dumping data for table `operadora_maquina`.`auto_atendimento`
--

/*!40000 ALTER TABLE `auto_atendimento` DISABLE KEYS */;
INSERT INTO `auto_atendimento`
(`cod_maquina`,`produto`,`tipo_produto`,`cod_autorizador`,`valor_produto`,`estoque`) VALUES
 ('000001','refrigerante','cola','1000000001','1.50','8'),
 ('000002','refrigerante','guaraná','1000000002','1.50','10'),
 ('000003','cafe','longo','1000000003','0.50','10'),
 ('000004','jornal','expresso','1000000004','2.00','8');
/*!40000 ALTER TABLE `auto_atendimento` ENABLE KEYS */;

--
-- Table structure for table `operadora_maquina`.`clientes`
--

DROP TABLE IF EXISTS `clientes`;
CREATE TABLE `clientes` (
  `nome` varchar(45) NOT NULL default '' COMMENT 'nome do cliente',
  `endereco` varchar(45) NOT NULL default '' COMMENT 'endereço do cliente',
  `msisdn` varchar(16) NOT NULL default '' COMMENT 'numero do celular do cliente',
  `categoria` varchar(10) NOT NULL default '' COMMENT 'categoria do celular',
  `saldo` decimal(8,2) NOT NULL default '0.00',
  `pendencia_financeira` tinyint(1) NOT NULL default '0',
  PRIMARY KEY  (`msisdn`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 COMMENT='Contem dados do Cliente da Operadora';

--
-- Dumping data for table `operadora_maquina`.`clientes`
--

/*!40000 ALTER TABLE `clientes` DISABLE KEYS */;
INSERT INTO `clientes` (`nome`,`endereco`,`msisdn`,`categoria`,`saldo`,`pendencia_financeira`) VALUES
 ('Andre Rabelo Gravina','AOS 5 BLOCO C APTO 306','556184010000','pos-pago','43.00',0),
 ('Leonardo PC','AOS 5 BLOCO B 200','556184010001','pre-pago','7.00',1),
 ('Francisco Javier','UniCEUB','556184010002','pre-pago','-10.00',1),
 ('Laila Vieira','Jardim Botânico','556184010003','pos-pago','13.00',1);
/*!40000 ALTER TABLE `clientes` ENABLE KEYS */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
```