

UNIVERSIDADE REGIONAL DE BLUMENAU
CENTRO DE CIÊNCIAS EXATAS E NATURAIS
CURSO DE SISTEMAS DE INFORMAÇÃO – BACHARELADO

**SISTEMA DE FORÇA DE VENDAS COM ATUALIZAÇÃO DE
PAGAMENTOS VIA WEB SERVICE**

GUSTAV DALLMANN JÚNIOR

BLUMENAU
2012

2012/2-15

GUSTAV DALLMANN JÚNIOR

**SISTEMA DE FORÇA DE VENDAS COM ATUALIZAÇÃO DE
PAGAMENTOS VIA WEB SERVICE**

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

Prof. Francisco Adell Péricas, Mestre - Orientador

**BLUMENAU
2012**

2012/2-15

SISTEMA DE FORÇA DE VENDAS

Por

GUSTAV DALLMANN JÚNIOR

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

Presidente: _____
Prof. Francisco Adell Péricas, Mestre – Orientador, FURB

Membro: _____
Prof. Rion Brattig Correia, Mestre – FURB

Membro: _____
Prof. Cláudio Ratke, Mestre – FURB

Blumenau, 30 de novembro de 2012.

Dedico este trabalho a todos os amigos,
especialmente aqueles que me ajudaram
diretamente na realização deste.

AGRADECIMENTOS

À minha família, que sempre esteve ao meu lado me dando todo o apoio necessário.

Aos meus amigos, pelas cobranças e ajuda.

Ao meu orientador, Francisco Adell Péricas, por ter acreditado na conclusão deste trabalho.

Aos professores do Departamento de Sistemas e Computação da Universidade Regional de Blumenau por suas contribuições durante os semestres letivos.

A conquista é dos persistentes e acontece batalha por batalha, dia a dia, e quanto mais difícil a luta, maior a recompensa.

Fabício S. Cruvinel

RESUMO

Este trabalho apresenta a realização de um sistema para a automação dos processos de uma microempresa que até o presente momento utiliza somente planilhas eletrônicas para controlar todas as suas operações diárias. Os focos principais do sistema são o controle das vendas, que faz-se necessário pois a empresa possui vendedores internos e externos, garantindo maior agilidade e segurança em suas ações e também a integração do sistema com o *web service* da operadora de crédito para que assim todos os recebimentos e pagamentos em aberto sejam sempre atualizados de forma rápida e segura.

Palavras-chave: *Web service*. Força de Vendas.

ABSTRACT

This work presents the realization of the automation from a small enterprise process that at the present time uses spreadsheets to control all daily operations. The main focus of the system is the control of sales, that is necessary because the small enterprise have internal and external sellers ensuring greater speed and security on his actions and also the system integration with the web service of the credit provider so that all receipts and the open payments are always updated quickly and safely.

Keyword: *Web service*. Sales force.

LISTA DE FIGURAS

Figura 1 - Sistema CoopCob	17
Figura 2 - Relatório de recebimentos	18
Figura 3 - Controle de pagamentos.....	18
Figura 4 - Controle das vendas no mês	19
Figura 5 - Sistema do acadêmico Anderson Germano	19
Figura 6 - Cadastro de pedido do trabalho correlato	20
Figura 7 - Diagrama de atividades.....	22
Figura 8 - Diagrama de caso de uso	24
Figura 9 – Diagrama de classes	25
Figura 10 - Ferramenta pgAdmin	27
Figura 11 - Hibernate na classe Vendas	28
Figura 12 - Funcionamento do JSP	29
Figura 13 - Operações nos web services	31
Figura 14 - Camadas conceituais de web services	32
Figura 15 - Tela de login	36
Figura 16 - Usuário não administrador.....	36
Figura 17 - Tela principal	37
Figura 18 - Cadastro de vendas	37
Figura 19 - Itens para a venda	38
Figura 20 - Botão - Boletos	38
Figura 21 - Adicionar boleto	39
Figura 22 - Dados para cadastro de boleto	40
Figura 23 - Boleto.....	40
Figura 24 - Lista de colaboradores	41
Figura 25 - Novo cadastro de colaborador	41
Figura 26 – Lista de produtos	42
Figura 27 - Novo cadastro de produto	42
Figura 28 – Lista de clientes.....	43
Figura 29 - Novo cadastro de cliente.....	43
Figura 30 - Atualização de boletos	44

Figura 31 - Tela de atualização de boletos	44
Figura 32 - WSDL do web service	45
Figura 33 - Interface "ServicosSoap"	46
Figura 34 - Interface "Servicos"	46
Figura 35 - Interface "ServicosLocator"	47
Figura 36 - Interface "ServicosSoapStub"	47
Figura 37 - Classe "UseWS"	48
Figura 38 - Boleto encontrado após consulta ao web service	49
Figura 39 - Dados do boleto atualizados	49

LISTA DE QUADROS

Quadro 1 - Requisitos funcionais	23
Quadro 2 - Requisitos não funcionais	23
Quadro 3 - <i>Login</i> no sistema (UC01)	54
Quadro 4 - Gerir clientes (UC02).....	54
Quadro 5 – Gerir produtos (UC03).....	55
Quadro 6 – Gerir colaborador (UC006)	55
Quadro 7 – Gerir vendas (UC04)	56
Quadro 8 – Gerir boletos (UC05).....	56
Quadro 9 - Atualização de boletos (UC017)	57
Quadro 10 - Tabela Colaborador	58
Quadro 11 - Tabela Venda.....	58
Quadro 12 - Tabela Boleto	59
Quadro 13 - Tabela Cliente	59
Quadro 14 - Tabela Item.....	59
Quadro 15 - Tabela Produto	60

LISTA DE SIGLAS

CRM - *Customer relationship management*

FURB – Fundação Universidade Regional de Blumenau

JSP – *Java Server Pages*

HTML - *HyperText Markup Language*

HTTP - *HyperText Transfer Protocol*

MER – Modelo Entidade Relacionamento

POP - *Post Office Protocol*

RF – Requisitos Funcionais

RNF – Requisitos Não Funcionais

SMTP - *Simple Mail Transfer Protocol*

SOAP - *Simple Object Access Protocol*

UDDI - *Universal Description, Discovery and Integration*

UML - *Unified Modeling Language*

WSDL - *Web Service Description Language*

XML - *Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO.....	12
1.1 OBJETIVOS DO TRABALHO	12
1.2 ESTRUTURA DO TRABALHO	13
2 FUNDAMENTAÇÃO TEÓRICA.....	14
2.1 FORÇA DE VENDAS	14
2.2 PRESTAÇÃO DE SERVIÇOS	15
2.2.1 CARACTERÍSTICAS DOS SERVIÇOS	16
2.3 SISTEMA ATUAL	17
2.4 TRABALHOS CORRELATOS	19
3 DESENVOLVIMENTO.....	21
3.1 LEVANTAMENTO DE INFORMAÇÕES.....	21
3.2 ESPECIFICAÇÃO	22
3.2.1 REQUISITOS FUNCIONAIS	23
3.2.2 REQUISITOS NÃO FUNCIONAIS	23
3.2.3 CASOS DE USO	23
3.2.4 DIAGRAMA DE CLASSES	24
3.3 IMPLEMENTAÇÃO	25
3.3.1 TÉCNICAS E FERRAMENTAS UTILIZADAS.....	26
3.3.1.1 POSTGRESQL 9.2.....	26
3.3.1.2 APACHE TOMCAT 6.0	27
3.3.1.3 JAVA SERVER PAGES	27
3.3.1.4 HIBERNATE	28
3.3.1.5 APACHE AXIS	29
3.3.1.6 <i>JAVA SERVER PAGES</i>	29
3.3.1.7 <i>WEB SERVICES</i>	30
3.3.1.7.1 CAMADAS CONCEITUAIS DE <i>WEB SERVICES</i>	32
3.3.1.7.2 SIMPLE OBJECT ACCESS PROTOCOL (SOAP)	33
3.3.1.7.3 WEB SERVICE DESCRIPTION LANGUAGE (WSDL)	34
3.3.1.7.4 UNIVERSAL DISCOVERY AND DESCRIPTION INTEGRATION (UDDI)	35
3.4 OPERACIONALIDADE DA IMPLEMENTAÇÃO	35
3.5 RESULTADOS E DISCUSSÃO	50

4 CONCLUSÕES.....	51
4.1 EXTENSÕES	51
REFERÊNCIAS BIBLIOGRÁFICAS	52
APÊNDICE A – DESCRIÇÃO DOS CASOS DE USO	54
APÊNDICE B – DICIONÁRIO DE DADOS.....	58

1 INTRODUÇÃO

O constante crescimento do mercado brasileiro reflete diretamente na necessidade de segurança dos dados de cada empresa. Evitar a perda de informações por conta de falhas de hardware ou software e garantir a integridade das informações é fundamental para o bom funcionamento das organizações.

A empresa Pereval Comércio de Purificação de Água Ltda situada na cidade de Blumenau atende também cidades vizinhas. Fatores como possuir pouca concorrência e estar há vários anos no mercado, aliado à conscientização da população com a melhoria da higiene e qualidade de vida fizeram com que a empresa crescesse rapidamente nos últimos anos.

Junto com o crescimento houve um aumento no volume das transações diárias e uma maior dificuldade no controle da empresa que sempre foi feito de forma manual por meio de planilhas eletrônicas.

Outro ponto impactante no controle da empresa é que todas as confirmações dos pagamentos recebidos são obtidas acessando-se um ambiente *online*, onde o administrador da empresa faz *download* dos recebimentos de hora em hora e a partir destes dados faz a atualização de suas planilhas eletrônicas.

Visando realizar a automação dos processos da empresa, a fim de garantir a segurança dos dados e também garantir agilidade em todas as tarefas, este trabalho propõe a criação de um sistema de força de vendas que não só controle todos os dados da empresa, mas que também possibilite aos seus vendedores realizarem os lançamentos de suas vendas aumentando sua produtividade e diminuindo o tempo entre a conclusão do pedido e o contato para que o agendamento referente à instalação do produto seja feito.

1.1 OBJETIVOS DO TRABALHO

O objetivo do trabalho proposto consiste em desenvolver um sistema *web* para o gerenciamento de força de vendas de uma microempresa com o a integração ao *web service* que contém suas movimentações bancárias, afim de garantir a atualização de todas as informações necessárias visando agilidade, segurança e integridade dos dados dos clientes e da empresa.

Os objetivos específicos do trabalho são:

- a) controlar os dados dos clientes e dos vendedores, os recebimentos, as vendas, as visitas e os produtos;
- b) desenvolver a integração com o *web service* da operadora de crédito;

1.2 ESTRUTURA DO TRABALHO

Este trabalho está disposto em quatro capítulos. No primeiro capítulo, é apresentada a introdução do assunto, os objetivos a serem alcançados com o desenvolvimento e a estrutura do trabalho.

O segundo capítulo apresenta a fundamentação teórica sobre conceitos do JSP e *web service*, prestação de serviços, sistema atual e os trabalhos correlatos.

No terceiro capítulo têm-se a descrição do ciclo de desenvolvimento do sistema, detalhes sobre a especificação e modelagem, técnicas e ferramentas utilizadas e a operacionalidade do sistema com os resultados e discussões.

No quarto capítulo apresenta-se a conclusão sobre os objetivos alcançados e sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda assuntos como *web service*, prestação de serviços, força de vendas, sistema atual e trabalhos correlatos.

2.1 FORÇA DE VENDAS

Com o avanço tecnológico e a necessidade do uso de sistemas para o controle empresarial que buscavam a garantia, praticidade, auxílio e a integridade dos dados de toda a organização, surgiram os primeiros sistemas de controle de vendas, também conhecidos como *Customer relationship management (CRM)*.

Quando o termo força de vendas surgiu no mercado, o seu conceito era, segundo Carvalho (1976), "um grupo de profissionais altamente preparados cujo objetivo é acelerar a venda final de produtos ou de serviços".

Com o crescimento do mercado o termo continuou sendo usado e graças ao constante crescimento de tecnologias de informação e da necessidade mercadológica, empresas que têm a necessidade de otimizar os processos de trocas de informações buscam cada vez mais melhorias e agilidade no processo das vendas visando a satisfação dos clientes.

Segundo Batista (2004), a automação de processos é qualquer aplicação da tecnologia da informação destinada a aumentar a produtividade dos trabalhadores. O processo de automação de vendas pode ser compreendido como um grande aliado para o aumento de produtividade da organização e também para o desempenho da atividade do trabalhador.

Segundo O'Brien (2002), os envolvidos na automação da força de vendas estão sendo equipados com *notebooks*, navegadores de rede e softwares de gerenciamento assim como estão conectados à *internet*, *extranets* e às *intranets* da empresa. Isso não só aumenta a produtividade pessoal dos vendedores, mas também agiliza muito a capacitação e análise de dados de venda, do campo para os gerentes da empresa. Por sua vez, permite a administração de marketing e vendas melhorar a prestação das informações e o suporte que ela fornece os seus vendedores.

Para Klein (2010), a automação de vendas traz eficiência, pois livra os processos das amarras, controles e papeladas que os tornam custosos. Conforme Cardoso e Gonçalves

(2001), quando implantado de forma adequada, o sistema de automação de força de vendas reduz a duração dos ciclos dos processos relacionados com o cliente, o desperdício, principalmente de tempo, e agrega valor à relação com o cliente. Além disso, a organização passa a ter uma visão melhor do cliente, passando a tomar parte nessa relação.

Peppers e Rogers (2004) salientam que o sistema de automação de força de vendas deve contemplar algumas funcionalidades principais, entre elas: controle de pedidos, gerência de contatos, configurações de produtos, planejamento de visitas, produtividade pessoal entre outras.

Visando a automação dos processos da empresa que atualmente são gerenciados por planilhas eletrônicas e buscando maior agilidade nos mesmos, verificou-se a necessidade do desenvolvimento de um sistema que contemple todas as funcionalidades necessárias para um bom funcionamento das atividades diárias.

2.2 PRESTAÇÃO DE SERVIÇOS

Segundo a Casa Civil (1990), o Art. 3º do código de defesa do consumidor define que serviço é qualquer atividade fornecida no mercado de consumo, mediante remuneração, inclusive as de natureza bancária, financeira, de crédito e securitária, salvo as decorrentes de caráter trabalhista.

Segundo Las Casas (2004, p.15), “serviços são atos, ações e desempenho. Esta é uma das melhores definições de serviços, uma vez que engloba de forma simples e objetiva todas as categorias de serviços, querem sejam eles agregados a um bem ou não. O produto final de um serviço é sempre um sentimento. Os clientes ficam satisfeitos ou não conforme suas expectativas. Portanto a qualidade em serviços é variável de acordo com o tipo de pessoa”.

Segundo Albrecht (1992), “qualidade em serviço é a capacidade que uma experiência ou qualquer outro fator tenha para satisfazer uma necessidade, resolver um problema ou fornecer benefícios a alguém”.

Segundo Moller (1997), “dentro de uma empresa a qualidade do serviço é definida como o grau até o qual um serviço satisfaz os requisitos descritos em suas especificações.”.

Tavares (2010), afirma que o ramo do setor de serviços é um dos que mais cresce atualmente no Brasil e no mundo.

Atualmente as empresas precisam assegurar que essas demandas e padrões, uma vez estabelecidos, sejam cuidadosamente atendidos. É fundamental para que em todas as empresas prestadoras de serviços, que esta relação de comprometimento seja mantida sempre e a qualquer custo, tendo em vista o desejo de manter seus clientes e de atendê-los da melhor forma possível.

2.2.1 CARACTERÍSTICAS DOS SERVIÇOS

Segundo Las Casas (2004, p 24-25), os serviços apresentam as seguintes características: são intangíveis, inseparáveis, heterogêneos e simultâneos:

- a) intangibilidade: a característica de intangibilidade significa que os serviços são abstratos;
- b) inseparabilidade: o aspecto da inseparabilidade dos serviços refere-se a outro importante determinante mercadológico de comercialização. Não se pode produzir ou estocar serviços como se fossem bens. Geralmente, os serviços são prestados quando vendedor e comprador estão frente a frente. Por isso é necessária uma capacidade de prestação de serviço antecipada;
- c) heterogeneidade: refere-se à impossibilidade de se manter a qualidade do serviço constante. Os serviços são produzidos pelo ser humano, que é por natureza instável. Sua qualidade da produção será também instável;
- d) simultaneidade: a simultaneidade dos serviços nos diz que produção e consumo ocorrem ao mesmo tempo e, sendo assim, será necessário sempre considerar o momento de contato com a clientela como fator principal de qualquer esforço mercadológico.

A prestação de serviços é entendida como a realização de trabalho oferecido ou contratado por terceiros (comunidade ou empresa), incluindo assessorias, consultorias e cooperação interinstitucional.

2.3 SISTEMA ATUAL

Atualmente a empresa Pereval Comércio de Purificação de Água utiliza unicamente planilhas eletrônicas para manter o seu controle de recebimentos, vendas, produtos e clientes, sendo o administrador da empresa o responsável pela manutenção das mesmas.

O controle dos recebimentos e da geração de boletos referente às vendas concluídas é feito através de um sistema *online* voltado para cobranças bancárias de cooperativas de crédito. Na Figura 1 tem-se o sistema *online* utilizado para a manutenção e controle dos dados da empresa.

The screenshot displays the web interface of the CoopCob system. At the top, the logo of 'COOPERATIVA DE CRÉDITO VALE DO ITAJAÍ' is visible, along with the text 'VIACREDI Com Você desde 1951'. The user is logged in as 'pereval.agua@yahoo.com.br'. The main menu includes 'Cadastros', 'Boletos', 'Pagamentos', 'Relatorios', 'Usuarios do Sistema', and 'Ajuda'. The current page is titled 'Relatórios - Extrato de Sacado'. The search criteria are set for the period from 01/11/2012 to 30/11/2012. The 'Sacado' section has radio buttons for 'Todos' (selected) and 'Único', followed by a search button 'Procurar'. Below that, there are radio buttons for 'Válidas' (selected) and 'Todas'. At the bottom of the search area, there are options to 'Exportar em PDF' and 'Exportar para MS-Excel' (selected), and a 'Visualizar Impressão' button. The footer of the page shows the 'Atoda Sistemas' logo.

Fonte: <https://www.coopcob.com.br/>

Figura 1 - Sistema CoopCob

Por meio deste sistema, o administrador da empresa tem todo o conteúdo que faz-se necessário para o controle da empresa. Para manter o controle atual da empresa sempre atualizado o administrador faz o *download* de um relatório de seus recebimentos de hora em hora, porém o arquivo gerado pelo sistema somente mostra alguns dados como o nome do cliente, o número da parcela, o valor da parcela, e as datas de pagamento e vencimento.

Conforme mostra a Figura 2, tem-se uma planilha Excel que contém o relatório de recebimentos de um determinado período.

PARC	OCORRÊNCIA	DATA	VALOR	DATA	VALOR
9/10	A receber	01/12/2011	68,00		0,00
4/6	Pagamento Dinho	09/11/2011	89,00	13/02/2012	89,00
5/3	Via Banco	05/11/2011	300,00	07/11/2011	300,00
5/15	Via Banco	05/11/2011	49,00	09/11/2011	49,33
1/1	Via Banco	05/11/2011	170,00	07/11/2011	170,00
1/1	Via Banco	03/11/2011	110,00	03/11/2011	110,00

Figura 2 - Relatório de recebimentos

A planilha eletrônica usada para o controle da empresa contém todos os dados de cada venda efetuada juntamente com o nome do vendedor responsável e o status de cada boleto gerado no momento da conclusão da venda, além de um controle mais detalhado dos pagamentos. Após a geração do relatório de recebimentos, os dados são copiados um a um para o controle da empresa.

Na Figura 3 tem-se o controle atual referente aos pagamentos recebidos dos clientes.

Pedido	Vend.	N° Bloq.	Bco	Ag	CH	Valor	N° Par	Cliente	Venc	Pagt	Endereço
N12310	Aristides	17805	EM	ESP	Camara	89,00	2	***	10/09/10	30/09/10	***
N11672	Roberto/Rona	VER		C.C	AJ	43,00	6	***	10/09/10	30/10/10	***
N7266	Aristides	72660910	EM	ESP	PL	50,00	9	***	10/03/09	30/09/10	***
01/ago	Wilson		EM	ESP	Camara	200,00	1	***	30/09/10	01/10/10	***
N10492	Alciano	5782		B.B	SMART	66,00	9	***	15/10/10	01/10/10	***
N389	Petri	7257		B.B	PL	69,00	10	***	08/09/10	01/10/10	***
N11103	Aristides	8710		B.B	SMART	58,00	10	***	12/10/10	01/10/10	***
N459	Petri	8748		B.B	PL	67,00	10	***	15/10/10	01/10/10	***
N11132	Aristides	9076		B.B	SMART	58,00	8	***	10/09/10	01/10/10	***
N11194	Aristides	9613		B.B	SMART	67,90	9	***	10/10/10	01/10/10	***
N12596	Aristides	20146		B.B	Camara	38,00	1	***	10/10/10	01/10/10	***
N11205	Lucas	10379		B.B	PL	53,00	8	***	03/10/10	01/10/10	***
N11464	Osmar	11764		B.B	FLEX	108,00	6	***	05/10/10	01/10/10	***
N11554	Aristides	12175		B.B	Camara	38,00	2	***	10/05/10	01/10/10	***

Figura 3 - Controle de pagamentos

Na Figura 4 tem-se o controle atual das vendas do mês corrente.

	A	B	C	D	E	F	G	H	I	J	K		
	Pedido	Vend.	N°	Bc	Ag	CH	Valor	N°	Par	Cliente	Venc	Pagt	Endereço
2	N12784	Aristides	21395		B.B	SMART	55,99	1		OLGA LANGE BARBIER	10/12/10		***
3	N12784	Aristides	21396		B.B	SMART	55,99	2		OLGA LANGE BARBIER	10/01/11		***
4	N12784	Aristides	21397		B.B	SMART	55,99	3		OLGA LANGE BARBIER	10/02/11		***
5	N12784	Aristides	21398		B.B	SMART	55,99	4		OLGA LANGE BARBIER	10/03/11		***
6	N12784	Aristides	21399		B.B	SMART	55,99	5		OLGA LANGE BARBIER	10/04/11		***
7	N12784	Aristides	21400		B.B	SMART	55,99	6		OLGA LANGE BARBIER	10/05/11		***
8	N12784	Aristides	21401		B.B	SMART	55,99	7		OLGA LANGE BARBIER	10/06/11		***
9	N12784	Aristides	21402		B.B	SMART	55,99	8		OLGA LANGE BARBIER	10/07/11		***
10	N12784	Aristides	21403		B.B	SMART	55,99	9		OLGA LANGE BARBIER	10/08/11		***
11	N12784	Aristides	21404		B.B	SMART	55,99	10		OLGA LANGE BARBIER	10/09/11		***
12	N12784	Aristides	21405		B.B	SMART	55,99	11		OLGA LANGE BARBIER	10/10/11		***
13	N12784	Aristides	21406		B.B	SMART	55,99	12		OLGA LANGE BARBIER	10/11/11		***
14	N12784	Aristides	21407		B.B	SMART	55,99	13		OLGA LANGE BARBIER	10/12/11		***
15	N12785	Aristides	21408		B.B	SMART	39,00	1		AILTON GOMES DA SILVA	10/11/10		***

Figura 4 - Controle das vendas no mês

2.4 TRABALHOS CORRELATOS

Germano (2003), em seu trabalho de conclusão do curso de Ciências da Computação pela Universidade Regional de Blumenau (FURB), ressalta que *web services* são aplicações modularizadas que podem ser descritas, publicadas e invocadas sobre uma rede, geralmente a WEB e que permitem uma integração de serviços de maneira rápida e eficiente. O autor implementou um sistema para consultas e alocações de recursos utilizando *web services*.

A Figura 5 apresenta a tela de visualização de requisição de recursos do trabalho de conclusão de curso do acadêmico Anderson Germano.

Visualização e requisição de recursos

Pesquisar Recursos Código Data cadastro

Verificar reservas Nome
 HOTEL DAS PALMEIRAS

Endereço CEP Telefone
 RUA XV DE NOVEMBRO 89046560 330-1634

Cadastrar clientes Cidade UF Cpl/Cgc
 BLUMENAU SC 000.000.000/0000-00

Cadastrar fornecedores URL
 www.hoteldaspalmeiras.com/wssserver

Procurar Fornecedores

Figura 5 - Sistema do acadêmico Anderson Germano

Machado (2007), em seu trabalho de conclusão do curso de Ciências da Computação pela Universidade Regional de Blumenau (FURB), ressalta que o uso de uma tecnologia bem formulada pode levar a empresa ao topo do ranking da competição mercadológica, ou seja, a tornará a concorrente de maior peso do mercado. A partir da necessidade de fornecer à força de vendas mobilidade e agilidade para a tomada de decisões em qualquer momento e lugar, assim fortalecendo o vínculo com os clientes e elevando seu índice de satisfação. O mesmo implementou um sistema gerencial para automação de força de vendas usando dispositivos móveis baseados em Palm OS.

A Figura 6 apresenta a tela de cadastro de pedidos do trabalho de conclusão de curso do acadêmico Tiago Machado.



Figura 6 - Cadastro de pedido do trabalho correlato

Tendo como fundamento os trabalhos mencionados a cima, neste trabalho de conclusão de curso pretende-se desenvolver um sistema de força de vendas utilizando-se *web service*, que servirá para o gerenciamento de uma microempresa e que se propõe a ser eficaz, de fácil manuseio e que atenda às necessidades do cliente em suas tarefas diárias visando maior agilidade em seus processos.

3 DESENVOLVIMENTO

Neste capítulo estão descritos os requisitos funcionais, os requisitos não funcionais, as regras de negócio, o diagrama de casos de uso, o fluxo de atividades com o software desenvolvido, o modelo de entidade relacionamento, a implementação, os resultados e a sua discussão.

3.1 LEVANTAMENTO DE INFORMAÇÕES

A solução desenvolvida surgiu através do contato com o administrador da empresa Pereval Comércio de Purificação de Água Ltda, que demonstrou como é o seu atual processo de trabalho. Hoje todo o controle da empresa é feito manualmente utilizando-se planilhas eletrônicas para garantir os dados dos clientes, das vendas, dos produtos e dos colaboradores. Como o controle é manual este processo é lento e propenso a falhas.

Optou-se por automatizar os processos da empresa em especial o de geração de vendas, afim de garantir que os dados de cada nova venda feita estejam disponíveis para a empresa logo após a sua conclusão.

Para que a solução desenvolvida ficasse completa também foram criados os controles dos dados dos clientes, dos produtos, dos colaboradores assim como também foi criado a integração com o *web service* da operadora de crédito a qual a empresa é cliente. Esta integração tem como objetivo a consulta dos pagamentos recebidos afim de atualizar o status dos boletos de cada venda paga.

Na Figura 7 é possível visualizar o fluxo atual da empresa.

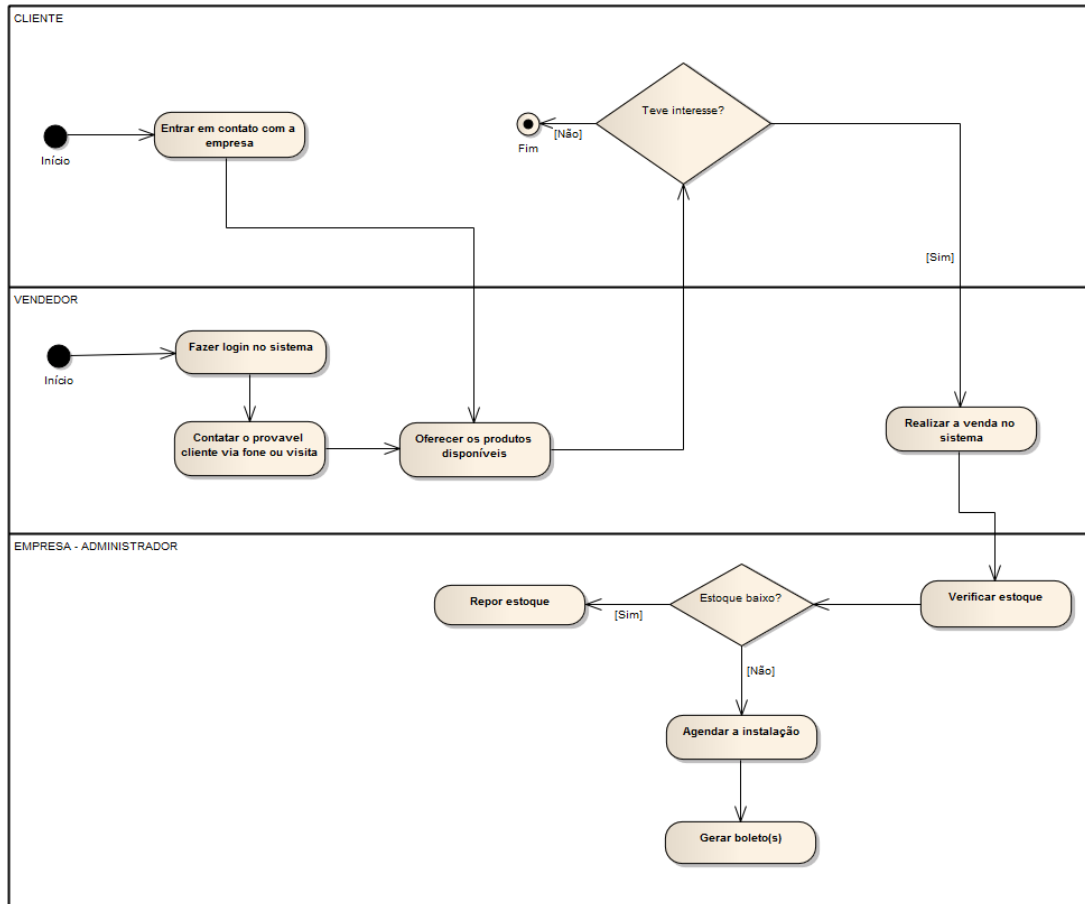


Figura 7 - Diagrama de atividades

A Figura 7 apresenta o diagrama de atividades que inicia no contato do cliente para com a empresa ou no contato do vendedor com o provável cliente até a finalização da venda. Neste diagrama podem ser verificadas todas as atividades a serem executadas no sistema, exceto a geração de novos boletos que ainda é feita pelo ambiente da operadora e crédito.

3.2 ESPECIFICAÇÃO

Nesta seção serão apresentados os principais requisitos funcionais (RF), requisitos não funcionais (RNF), sua rastreabilidade com casos de uso e o modelo entidade relacionamento (MER) do sistema. A descrição dos casos de uso pode ser visualizada no Apêndice A.

3.2.1 REQUISITOS FUNCIONAIS

No Quadro 1 são apresentados os requisitos funcionais do sistema.

Requisitos Funcionais	Caso de Uso
RF01: O sistema deverá efetuar <i>login</i> por perfil de usuário.	UC01
RF02: O sistema deverá gerir os dados de clientes.	UC02
RF03: O sistema deverá gerir os dados dos produtos.	UC03
RF04: O sistema deverá gerir os dados das vendas.	UC04
RF05: O sistema deverá gerir os dados dos boletos.	UC05
RF06: O sistema deverá gerir os dados dos colaboradores.	UC06
RF07: O sistema deverá permitir a atualização dos boletos já pagos.	UC07

Quadro 1 - Requisitos funcionais

3.2.2 REQUISITOS NÃO FUNCIONAIS

No Quadro 2 são apresentados os requisitos não funcionais do sistema.

Requisitos Não Funcionais
RNF01: O sistema deve ser desenvolvido na linguagem Java.
RNF02: O sistema deverá utilizar bando de dados PostgreSQL.
RNF03: O sistema deverá ser desenvolvido para plataforma <i>Web</i> .

Quadro 2 - Requisitos não funcionais

3.2.3 CASOS DE USO

Na Figura 8 pode-se verificar os casos de uso do sistema. Os atores envolvidos nestes casos de uso são o usuário e o administrador do aplicativo, onde o administrador pode cadastrar novos usuários e atualizar boletos além de utilizar as funções do usuário normal.

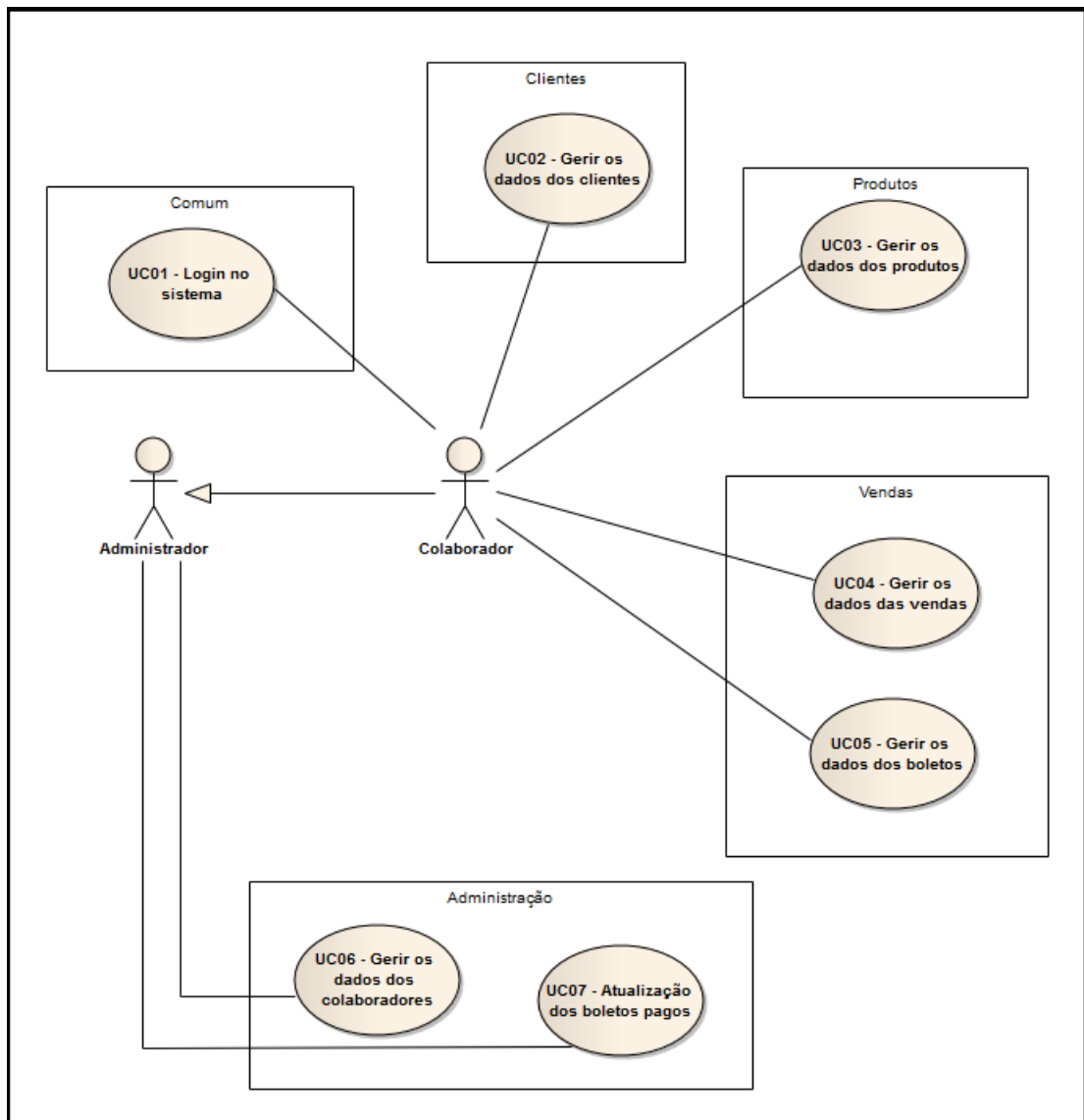


Figura 8 - Diagrama de caso de uso

3.2.4 DIAGRAMA DE CLASSES

Na Figura 9 verifica-se o diagrama de classes do sistema desenvolvido com as entidades criadas e seus relacionamentos. No Apêndice B pode-se visualizar o dicionário de dados.

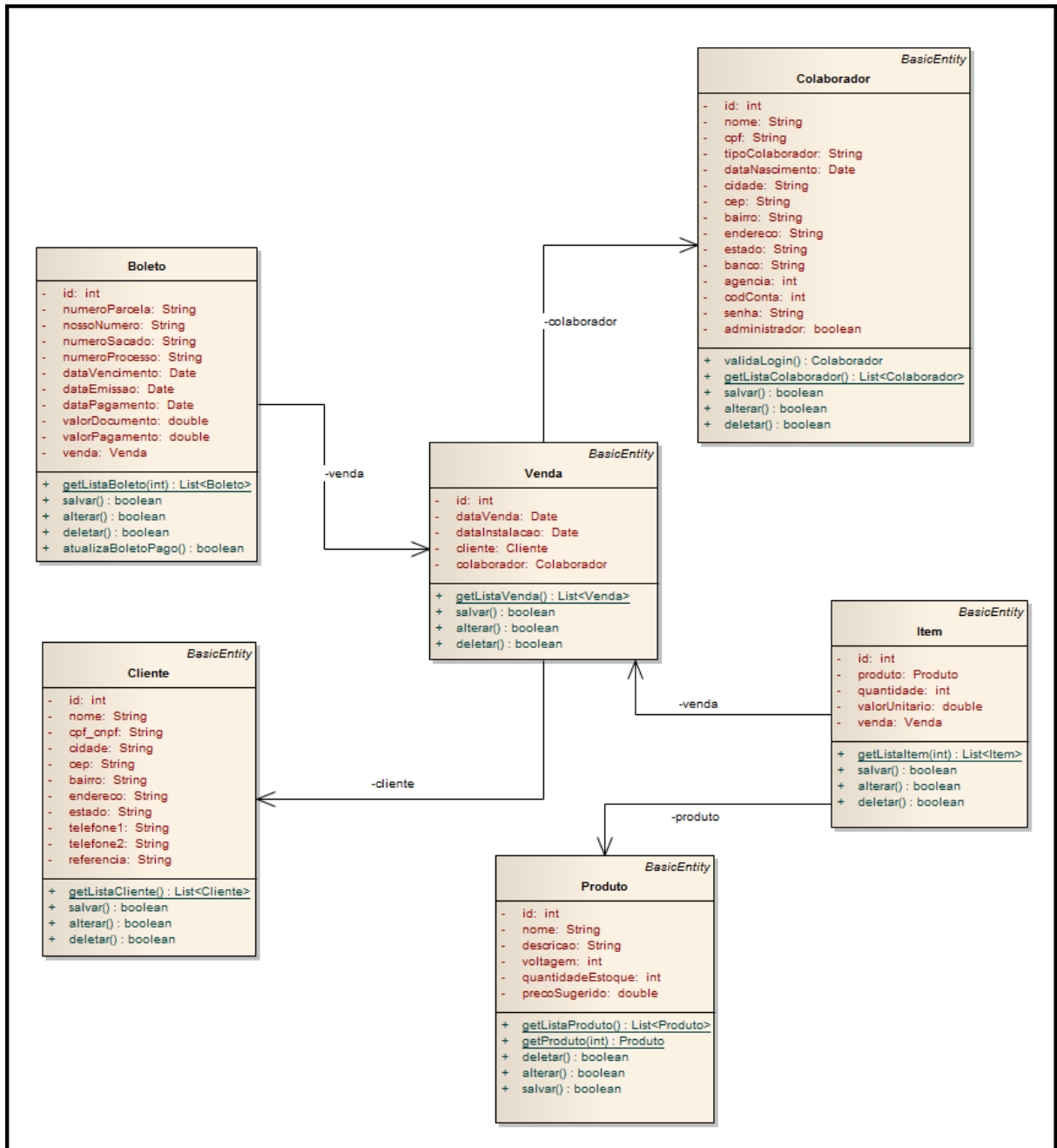


Figura 9 – Diagrama de classes

3.3 IMPLEMENTAÇÃO

A seguir são mostradas as técnicas e ferramentas utilizadas e a operacionalidade da implementação.

3.3.1 TÉCNICAS E FERRAMENTAS UTILIZADAS

A seguir são mostradas as técnicas e ferramentas utilizadas e a operacionalidade da implementação.

3.3.1.1 POSTGRESQL 9.2

A ferramenta de banco de dados utilizada para o sistema desenvolvido foi o dbExperts PostgreSQL 9.2. Algumas das vantagens oferecidas pelo uso do PostgreSQL são:

- a) redução de custos: é um banco de dados extremamente avançado, que não requer novos investimentos na infraestrutura existente, preservando o capital da empresa, pois dispensa os infundáveis ciclos de upgrade de software e de hardware;
- b) licença por servidor, sem limite de usuários: o licenciamento é por servidor, sem complicações;
- c) solução completa para Internet: acompanha o pacote um conjunto completo de ferramentas para desenvolvimento de aplicações para internet, bem como servidores *web* e servidores de aplicações;
- d) multiplataforma: O PostgreSQL é a única versão no mundo que funciona no ambiente Windows 95/98/ME/NT/2000/XP, operando nativamente como um serviço. Da mesma forma também suporta a plataforma Mac OS X, FreeBSD e Linux;
- e) migração facilitada: Migrações de outros bancos de dados são extremamente simples. O pacote inclui diversos aplicativos que tornam as etapas de migração fáceis e rápidas, economizando tempo.

Para o gerenciamento do banco de dados foi utilizada a ferramenta pgAdmin III, conforme apresentado na Figura 10.

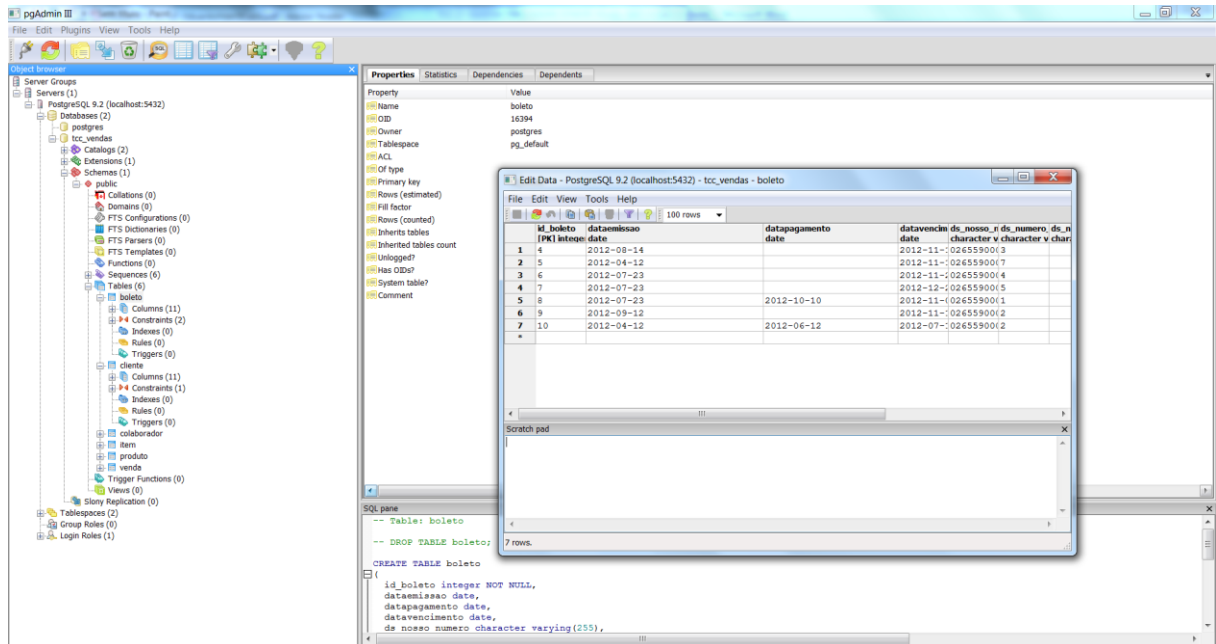


Figura 10 - Ferramenta pgAdmin

3.3.1.2 APACHE TOMCAT 6.0

Para a execução da ferramenta proposta optou-se pelo servidor web com base nas vantagens que oferece.

O Tomcat é um servidor de aplicações Java para *web*. É software livre e de código aberto, surgido dentro do conceituado projeto Apache Jakarta e que teve apoio e endosso oficial da Sun Microsystems como Implementação de Referência (RI) para as tecnologias Java Servlet e JavaServer Pages (JSP). Atualmente, o Tomcat tem seu próprio projeto de desenvolvimento independente, dentro da Apache Software Foundation. O Tomcat é robusto e eficiente o suficiente para ser utilizado mesmo em um ambiente de produção. (APACHE, 2011).

3.3.1.3 JAVA SERVER PAGES

O *Java Server Pages* (JSP) é uma tecnologia utilizada para o desenvolvimento de aplicações Web. Conforme Pittella (2006), arquivos JSP contêm HTML junto com o código embutido que permitem que o designer de páginas acessem dados do código Java

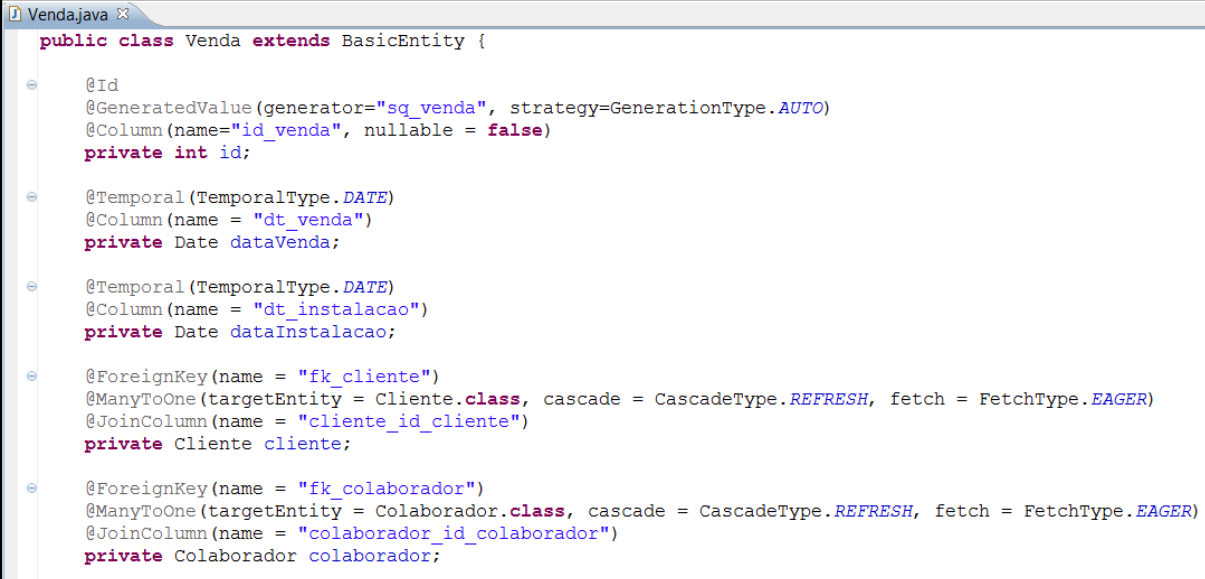
rodando no servidor. Quando a página é solicitada por um usuário e processada pelo servidor HTTP, a parte HTML da página é então transmitida.

No entanto, as partes de código das páginas são executadas no momento em que a solicitação é recebida e o conteúdo dinâmico gerado por este código é unido na página, antes de ser enviado para o usuário. Isto propicia uma separação dos aspectos de apresentação HTML da página, da lógica de programação contida no código.

3.3.1.4 HIBERNATE

Segundo Oliveira (2003), Hibernate é um mecanismo simples e poderoso que permite a persistência de objetos em banco de dados relacionais de maneira transparente e para qualquer tipo de aplicação Java.

Na Figura 11 apresenta a utilização do Hibernate na classe de Vendas.



```

public class Venda extends BasicEntity {

    @Id
    @GeneratedValue(generator="sq_venda", strategy=GenerationType.AUTO)
    @Column(name="id_venda", nullable = false)
    private int id;

    @Temporal(TemporalType.DATE)
    @Column(name = "dt_venda")
    private Date dataVenda;

    @Temporal(TemporalType.DATE)
    @Column(name = "dt_instalacao")
    private Date datainstalacao;

    @ForeignKey(name = "fk_cliente")
    @ManyToOne(targetEntity = Cliente.class, cascade = CascadeType.REFRESH, fetch = FetchType.EAGER)
    @JoinColumn(name = "cliente_id_cliente")
    private Cliente cliente;

    @ForeignKey(name = "fk_colaborador")
    @ManyToOne(targetEntity = Colaborador.class, cascade = CascadeType.REFRESH, fetch = FetchType.EAGER)
    @JoinColumn(name = "colaborador_id_colaborador")
    private Colaborador colaborador;
}

```

Figura 11 - Hibernate na classe Vendas

Na prática, ao invés de perder tempo escrevendo SQL o usuário vai precisar somente se preocupar com seus objetos. Além disso, o processo de desenvolvimento usando Hibernate é muito mais simples do que usar JDBC puro. Com Hibernate, o usuário só precisa se preocupar com cinco itens, sendo eles:

- a) criar a tabela no banco de dados onde os objetos vão persistir;
- b) criar o objeto cujo estado vai ser persistido;

- c) criar um XML, que relaciona as propriedades do objeto aos campos na tabela
- d) criar um arquivo contendo as propriedades para que o Hibernate se conecte ao bando de dados;
- e) criar a classe DAO que vai persistir o objeto.

3.3.1.5 APACHE AXIS

Segundo Medeiros (2007), Axis é um *framework* que possibilita com enorme facilidade a criação de *web services*, uma ferramenta simples de ser utilizada e muito eficiente em seu objetivo.

O Apache Axis também pode ser compreendido como um conjunto de ferramentas para criação de *web services* e também para a comunicação e consumo dos mesmos e que tem como objetivo fazer todo ou quase todo trabalho de criação do SOAP.

Dentre as funcionalidades que o framework obtém as que mais se destacam são:

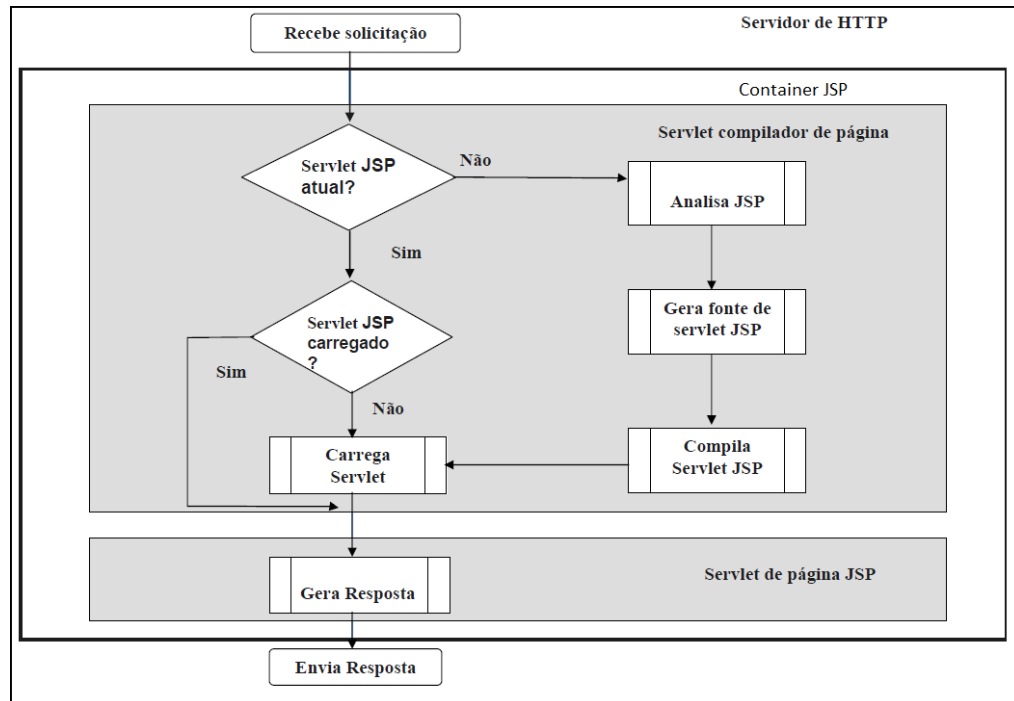
- a) implementação do protocolo SOAP;
- b) implementação de classes para agilizar a comunicação e a publicação de *web services*;
- c) receber e processar mensagens SOAP;
- d) criar um *web service* a partir de uma classe Java simples;
- e) facilidade para recuperar o WSDL de um serviço.

O Apache Axis é um projeto de código aberto, mantido pela Apache Software Foundation.

3.3.1.6 JAVA SERVER PAGES

O *Java Server Pages* (JSP) é uma tecnologia utilizada para o desenvolvimento de aplicações Web. Conforme Pittella (2006), arquivos JSP contêm HTML junto com o código embutido que permitem que o designer de páginas acessem dados do código Java rodando no servidor. Quando a página é solicitada por um usuário e processada pelo servidor HTTP, a parte HTML da página é então transmitida.

No entanto, as partes de código das páginas são executadas no momento em que a solicitação é recebida e o conteúdo dinâmico gerado por este código é unido na página, antes de ser enviado para o usuário. A Figura 12 apresenta este processo.



Fonte: Menin (2005).

Figura 12 - Funcionamento do JSP

Isto propicia uma separação dos aspectos de apresentação HTML da página, da lógica de programação contida no código.

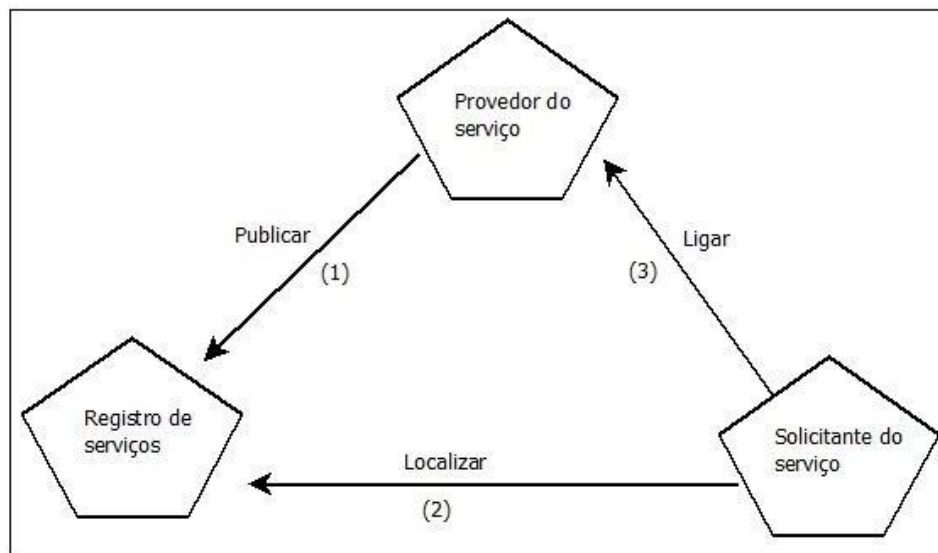
3.3.1.7 WEB SERVICES

Segundo Parreiras (2011), *web service* é um conceito que visa integrar aplicações mesmo que desenvolvidas em plataformas diferentes, trazendo assim agilidade e eficiência para os processos.

Seu principal objetivo é a comunicação de aplicações através da Internet. Esta comunicação é realizada de forma que haja interoperabilidade entre a informação que circula numa organização nas diferentes aplicações.

Segundo Kreger (2001), *web services* possuem uma arquitetura dividida em três funções (Figura 1):

- a) provedor de serviços: da perspectiva do negócio este é o dono do serviço. Da perspectiva arquitetural esta é a plataforma acessada na solicitação do serviço. É a entidade que cria o *web service* sendo responsável por fazer sua descrição em algum formato padrão e publicar os detalhes em um registro central;
- b) solicitante de serviços: é uma aplicação que invoca ou inicializa uma interação com o serviço. Pode ser um browser ou um programa sem interface com o usuário como por exemplo outro *web service*;
- c) registro de serviços: é a localização central onde o provedor de serviços pode relacionar seus *web services* e no qual um consumidor de serviços pode pesquisá-los obtendo as informações de ligação da descrição do serviço.



Fonte: Kreger (2001).

Figura 13 - Operações nos *web services*.

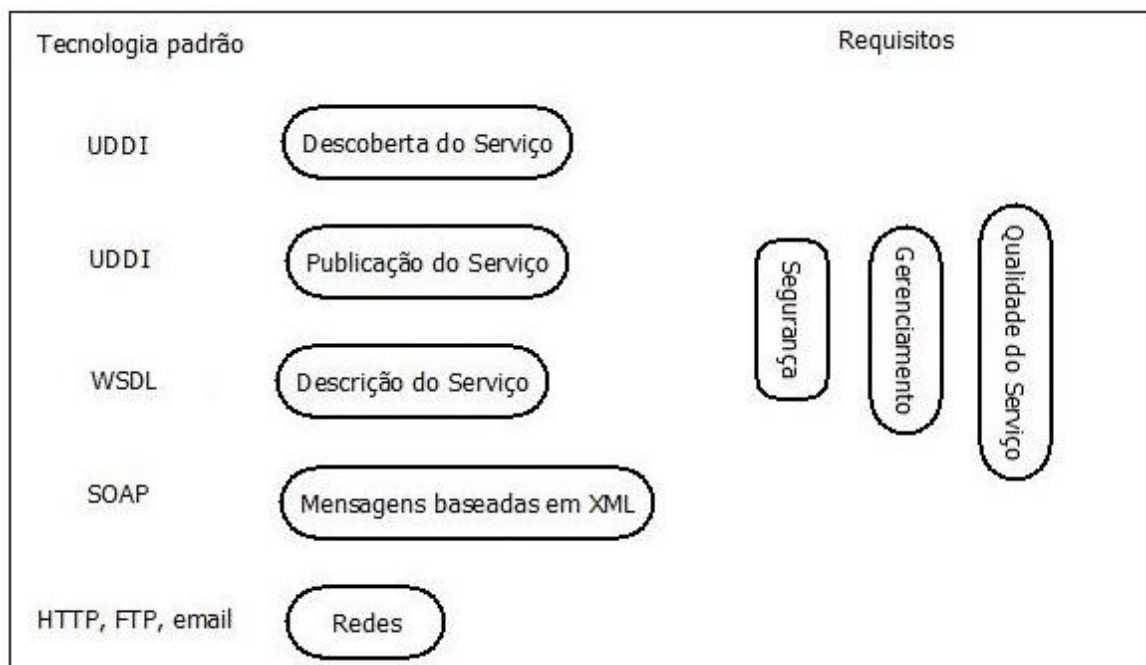
Na arquitetura dos *web services* apresentada na Figura 13, também existem três entidades que interagem entre si através das seguintes operações:

- a) ligar: quando um serviço necessita ser utilizado, a operação Ligar invoca e inicializa a interação com o mesmo em tempo de execução, usando as informações das ligações da descrição do serviço para localizar e contatá-lo;
- b) publicar: para que um serviço possa ser acessado, este deve ser publicado no Registro de Serviço. O Provedor de Serviço contata o Registro de Serviço para publicar um serviço;

- c) localizar: um Solicitante de Serviços encontra uma descrição do serviço ou consulta o Registro de Serviço para o tipo do serviço requerido. Um Solicitante de Serviços pode encontrar uma descrição da interface do serviço para o desenvolvimento de software na fase de design ou em tempo de execução. Desta forma, são encontradas as informações sobre ligações e localizações necessárias para invocar um serviço.

3.3.1.7.1 CAMADAS CONCEITUAIS DE *WEB SERVICES*

Para que seja possível uma fácil integração entre os componentes da arquitetura de *web services*, a colaboração entre eles está baseada no uso de protocolos padronizados. Estes protocolos são baseados em *Extensible Markup Language* (XML), destacando-se entre eles como os mais utilizados o *Hypertext Transfer Protocol* (HTTP), o *Simple Object Access Protocol* (SOAP), o *Web Service Description Language* (WSDL) e o *Universal Description, Discovery and Integration* (UDDI). A Figura 14 demonstra as camadas de um *web service*.



Fonte: Kreger (2001).

Figura 14 - Camadas conceituais de *web services*.

A Figura 14 demonstra as camadas conceituais dos *web services* (KREGER, 2001):

- a) camada de rede: é a camada base que representa protocolos como: *HyperText Transfer Protocol Secure* (HTTP), *File Transfer Protocol* (FTP), *Simple Mail Transfer Protocol* (SMTP), *Post Office Protocol* (POP), etc. É utilizada de acordo com os requisitos da aplicação: segurança, disponibilidade, performance e confiabilidade;
- b) mensagens baseada em *Extensible Markup Language* (XML): usa como padrão de tecnologia do protocolo *Simple Object Access Protocol* (SOAP) para troca de informações em um ambiente descentralizado e distribuído;
- c) descrição do serviço: a descrição do serviço é feita utilizando-se o *Web Service Description Language* (WSDL), que define uma interface e mecanismos de interação dos serviços além de descrições adicionais como contexto, qualidade do serviço e o relacionamento de serviço para serviços;
- d) publicação e descoberta do serviço: estas duas camadas utilizam a *Universal Discovery And Description Integration* (UDDI) para descoberta e publicação de informações sobre *web services*.

3.3.1.7.2 SIMPLE OBJECT ACCESS PROTOCOL (SOAP)

Consiste em um protocolo para troca de informações estruturadas em uma plataforma descentralizada e distribuída utilizando tecnologias baseadas em *Extensible Markup Language* (XML). É utilizado para invocar aplicações remotas ou *web services* através de chamadas remotas de procedimentos ou troca de mensagens em ambiente independente de plataforma e linguagem de programação (KREGGER, 2001).

De acordo com Costa (2007), a estrutura do protocolo SOAP é dividida em três itens, sendo eles:

- a) *Envelope*: Toda mensagem SOAP deve contê-lo. É o elemento raiz do documento XML. O *Envelope* pode conter declarações de *namespaces* e também atributos adicionais como o que define o estilo de codificação. Um "*encoding style*" define como os dados são representados no documento XML.

- b) *Header*: É um cabeçalho opcional. Ele carrega informações adicionais, como por exemplo, se a mensagem deve ser processada por um determinado nó intermediário (É importante lembrar que, ao trafegar pela rede, a mensagem normalmente passa por diversos pontos intermediários, até alcançar o destino final). Quando utilizado, o *Header* deve ser o primeiro elemento do *Envelope*.
- c) *Body*: Este elemento é obrigatório e contém o *payload*, ou a informação a ser transportada para o seu destino final. O elemento *Body* pode conter um elemento opcional *Fault*, usado para carregar mensagens de status e erros retornadas pelos "nós" ao processarem a mensagem.

3.3.1.7.3 WEB SERVICE DESCRIPTION LANGUAGE (WSDL)

Consiste em um documento *Extensible Markup Language* (XML), que descreve um conjunto de mensagens *Simple Object Access Protocol* (SOAP) e a forma como essas mensagens são trocadas.

O *Web Service Description Language* foi definido como sendo a linguagem padrão para descrever a interface e localização de um serviço. Especificamente o *WSDL* fornece um número de peças-chave de informação, sendo elas (KREGGER, 2001):

- a) a definição do formato das mensagens que são passadas entre dois pontos finais usando seus elementos `<type>` e `<message>` e definições apropriadas de esquemas (schema);
- b) a semântica do serviço: como ele deve ser chamado para fazer uma transmissão síncrona de solicitação e resposta, síncrona apenas de resposta ou comunicação assíncrona;
- c) o ponto final e o transporte do serviço através do elemento `<service>`: isto é, quem fornece o serviço;
- d) uma ligação através do elemento `<binding>`: isto é, como o serviço é alcançado.

Segundo Cunha (2007), quando o cliente deseja enviar uma mensagem para um determinado *web service*, ele obtém a descrição do serviço (através da localização do respectivo documento WSDL), e em seguida constrói a mensagem, passando todos os

parâmetros de acordo com a definição encontrada no documento. Em seguida, a mensagem é enviada para o endereço onde o serviço está localizado, a fim de que possa ser processada. O *web service*, quando recebe esta mensagem valida-a conforme as informações contidas no documento WSDL. À partir de então, o serviço remoto sabe como tratar a mensagem, sabe como processá-la (possivelmente enviando-a para outro programa) e como montar a resposta ao cliente.

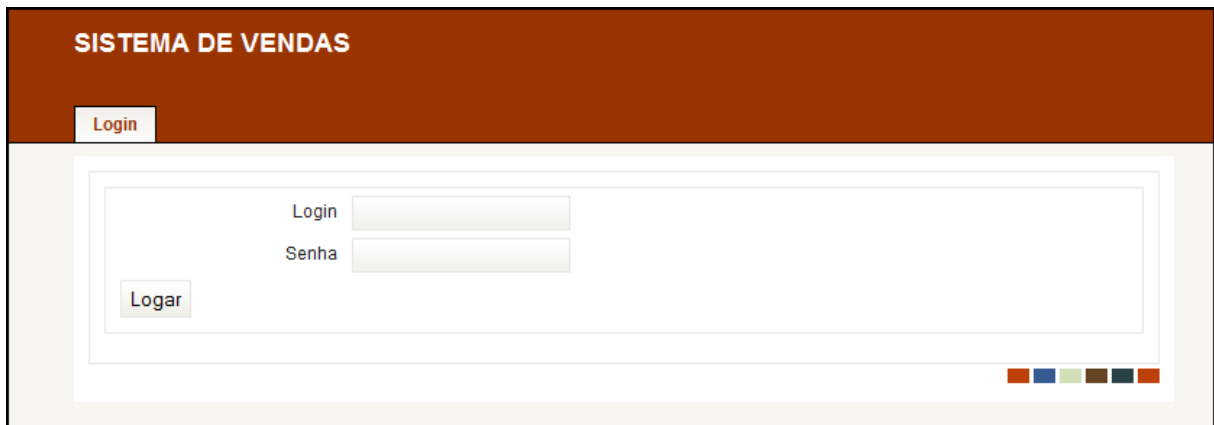
3.3.1.7.4 UNIVERSAL DISCOVERY AND DESCRIPTION INTEGRATION (UDDI)

O *Universal Discovery And Description Integration* (UDDI) é o mecanismo usado para publicar os serviços. A comunicação é realizada através do *Simple Object Access Protocol* (SOAP) e as interfaces *web service* são descritas pelo *Web Service Description Language* (WSDL).

Para fazer uma chamada a um *web service*, é necessário localizá-lo, descobrir a interface e semântica da sua chamada, escrever e configurar o software local para colaborar com o serviço.

3.4 OPERACIONALIDADE DA IMPLEMENTAÇÃO

A operacionalidade do sistema é inicialmente apresentada pela tela de *login*, onde o usuário do sistema deve preencher o campo de *login* e senha, como é apresentado na Figura 15. O campo *login* deve ser preenchido de acordo com o cadastro de pessoa física (CPF) de cada colaborador. Caso o usuário esqueça a senha deverá solicitar ao administrador da empresa que seja cadastrada uma nova senha.



The image shows a web application interface for 'SISTEMA DE VENDAS'. At the top, there is a dark brown header with the text 'SISTEMA DE VENDAS' in white. Below the header, there is a white box containing a login form. The form has two input fields: 'Login' and 'Senha'. To the left of the 'Senha' field is a 'Logar' button. In the bottom right corner of the white box, there are five small colored squares (orange, blue, green, brown, dark green).

Figura 15 - Tela de *login*

Após a realização do *login* no sistema, todo usuário é direcionado para tela de vendas, porém existem funcionalidades que são restringidas de acordo com o perfil do usuário logado. As opções de cadastro de funcionários e de atualização de boletos somente podem ser acessados pelo administrador da empresa. Caso outro usuário tente acessar estas opções o sistema irá mostrar um alerta em tela informando que o mesmo não possui permissão para isto, conforme mostra a Figura 16.

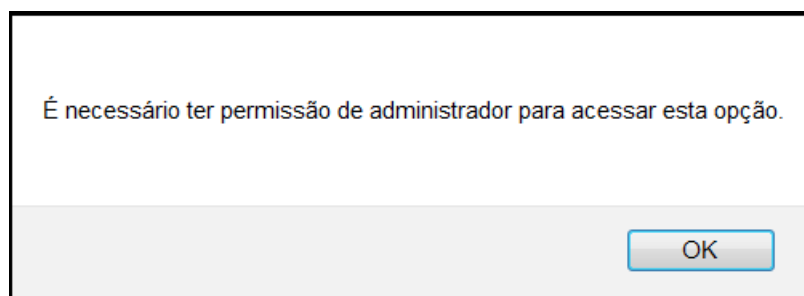


Figura 16 - Usuário não administrador

Conforme mostra a Figura 17, o sistema aponta para a tela de vendas logo após o usuário fazer o *login*. Nela estão dispostas todas as vendas lançadas no sistema permitindo ao usuário fazer novas inclusões ou alterações e exclusões conforme necessário.



Figura 17 - Tela principal

Ao clicar na opção para o cadastro de uma nova venda o sistema abre a tela de cadastro, conforme a Figura 18.

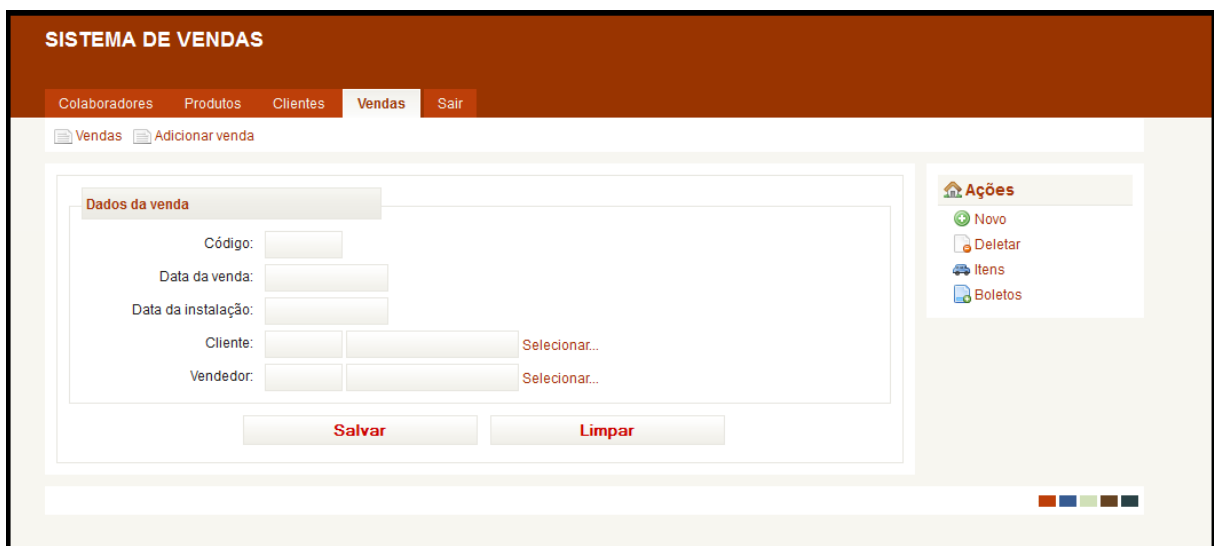


Figura 18 - Cadastro de vendas

Após o usuário informar os dados da venda e clicar em salvar, o mesmo deve fazer a inclusão dos produtos comprados pelo cliente.

Ao clicar na opção “Itens”, o sistema abre uma tela que contém uma lista do que está incluído na venda, caso ela já esteja concluída. Caso o usuário deseje vincular novos produtos à venda o mesmo deve clicar na opção “Adicionar item”. Neste momento o sistema abre a tela que possibilita a seleção dos produtos já cadastrados conforme mostra a Figura 19.

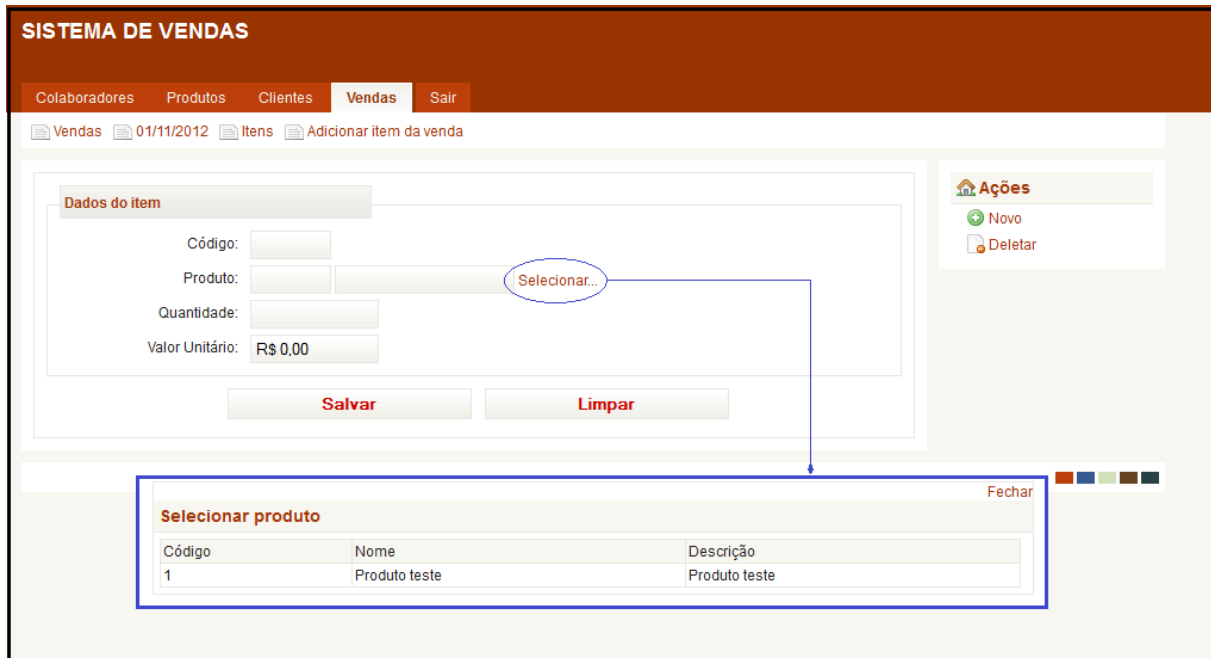


Figura 19 - Itens para a venda

Após clicar em “Salvar” o usuário concluiu os procedimentos necessários para a realização de uma nova venda. Caso o usuário seja o administrador da empresa, o mesmo possui permissão para adicionar os boletos gerados à venda. Para tal, na tela dos dados iniciais da venda existe o botão “Boletos”, conforme mostra a Figura 20.

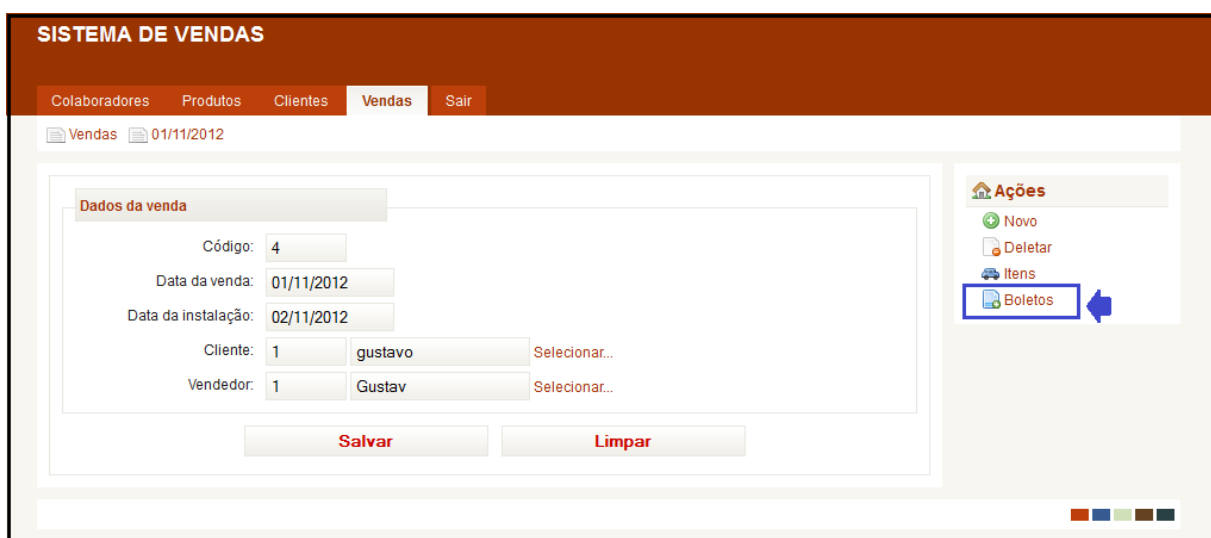
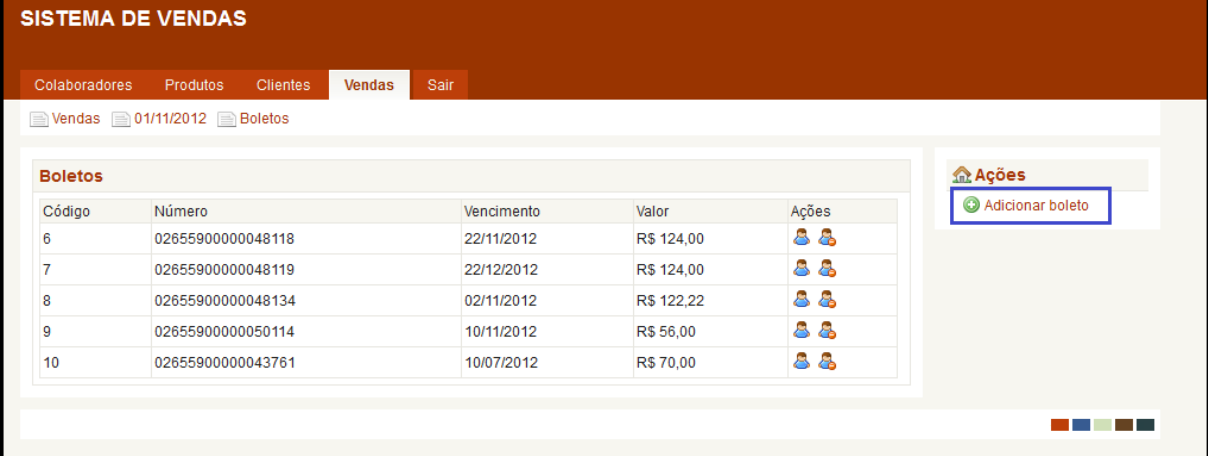












Figura 20 - Botão - Boletos

Ao se clicar nessa opção, o usuário tem acesso a todos os boletos existentes ou caso seja uma nova venda tem-se acesso ao cadastro de novos boletos por meio do botão “Adicionar boleto”, conforme mostra a Figura 21.



The screenshot displays the 'SISTEMA DE VENDAS' interface. At the top, there is a navigation menu with options: 'Colaboradores', 'Produtos', 'Clientes', 'Vendas', and 'Sair'. Below the menu, there are breadcrumb links: 'Vendas', '01/11/2012', and 'Boletos'. The main content area features a table titled 'Boletos' with the following data:

Código	Número	Vencimento	Valor	Ações
6	02655900000048118	22/11/2012	R\$ 124,00	 
7	02655900000048119	22/12/2012	R\$ 124,00	 
8	02655900000048134	02/11/2012	R\$ 122,22	 
9	02655900000050114	10/11/2012	R\$ 56,00	 
10	02655900000043761	10/07/2012	R\$ 70,00	 

To the right of the table, there is a sidebar with a section titled 'Ações' containing a button labeled 'Adicionar boleto' with a green plus icon. At the bottom right of the interface, there are four small colored squares: orange, blue, green, and black.

Figura 21 - Adicionar boleto

Em seguida o sistema abre a tela para cadastrar os dados do(s) boleto(s) gerado(s). Nesta tela o administrador necessita preencher os campos com os dados de cada boleto que foi gerado, conforme mostra a Figura 22.

SISTEMA DE VENDAS

Colaboradores Produtos Clientes **Vendas** Sair

Vendas 01/11/2012 Boletos Adicionar boleto da venda

Dados do boleto

Código:

Parcela: 4

Nosso número: 0265590000048118

Data emissão: 23/07/2012

Data vencimento: 22/11/2012

Valor do boleto: R\$ 124,00

Dados do pagamento

Valor pago: R\$ 0,00

Data pagamento:

Salvar Limpar

Ações

Novo Deletar

Figura 22 - Dados para cadastro de boleto

Os dados para o preenchimento destes campos são retirados dos boletos gerados no ambiente da ViaCredi, conforme mostra a Figura 23.

BANCO DO BRASIL	001-9	BANCO DO BRASIL	001-9	00194.57597 50265.590005 00048.118210 8 55250000012400
PARCELA 4/5	VENCIMENTO 22/11/2012	LOCAL DE PAGAMENTO ATÉ VENCTO PAGÁVEL EM QUALQUER BANCO. APÓS. SOMENTE NA VIACREDI OU BANCO DO BRASIL	PARCELA 4/5	VENCIMENTO 22/11/2012
AGÊNCIA / CÓDIGO DO CEDENTE 3420-7 / 2400-7	NOSSO NÚMERO 0265590000048118	CEDENTE PEREVAL COM DE PURIFICADORES DE AGUA LTDA	AGÊNCIA / CÓDIGO DO CEDENTE 3420-7 / 2400-7	NOSSO NÚMERO 0265590000048118
ESPECIE REAL	QUANTIDADE 124	DATA DO DOCUMENTO 23/7/2012	NR DO DOCUMENTO 0000048118	ESPECIE DOC DM
NR DO DOCUMENTO 0000048118	VALOR DO DOCUMENTO 124.00	DATA DO DOCUMENTO 23/07/2012	ESPECIE REAL	QUANTIDADE 124
DESCONTO / ABATIMENTO	OUTRAS DEDUÇÕES	INSTRUÇÕES APÓS VENCTO MULTA DE 1.00% (1.24) E JUROS DE MORA 2.00% AM (08 AO DIA).	VALOR DO DOCUMENTO 124.00	VALOR 124.00
MORA / MULTA	OUTROS ACRESCIMOS	PAGUE PREFERENCIALMENTE NA VIACREDI após vencimento somente no Banco Brasil ou Viacred. Banco Brasil ou Viacred, receber mesmo que ultrapasse os 30 dd.	DESCONTO / ABATIMENTO	DESCONTO / ABATIMENTO
VALOR COBRADO	Recibo do Sacado	SACADO CEP-89020300 BLUMENAU - SC	OUTRAS DEDUÇÕES	MORA / MULTA
	SACADO: A.P.F DO C.E.I ANTONIO JOSE CURTIPASZI CEDENTE: PEREVAL COM DE PURIFICADORES DE AGUA LTDA		OUTROS ACRESCIMOS	VALOR COBRADO
			VALOR COBRADO	Ficha de Compensação

Figura 23 - Boleto

Toda vez que houver admissões por parte da empresa, o administrador da empresa deve fazer o cadastro do novo colaborador no sistema. Para isto, ele deve acessar o menu “Colaboradores”. Neste momento o sistema apresenta todos os colaboradores cadastrados no sistema, conforme a Figura 24. É nesta tela que o administrador pode fazer alterações nos itens cadastrados clicando na ação “Mostrar” ou fazer a exclusão dos mesmos clicando na ação “Apagar”.

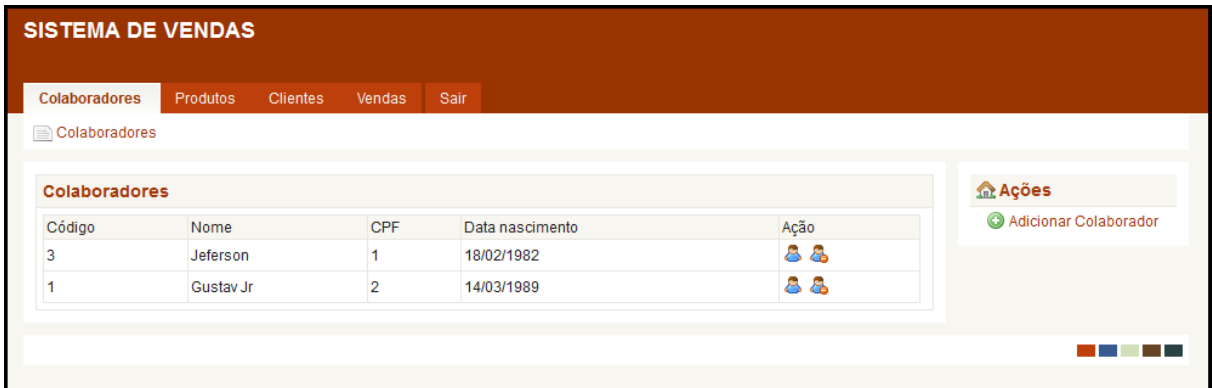


Figura 24 - Lista de colaboradores

Para o cadastro de um novo colaborador, deve-se clicar na opção “Adicionar colaborador”. Neste momento o sistema abre a tela que contém os dados necessários de cada novo funcionário. A tela de cadastro de colaboradores possui campos obrigatórios, conforme mostra a Figura 25.

Figura 25 - Novo cadastro de colaborador

Após o preenchimento dos campos, o administrador deve clicar em “Salvar” para concluir o cadastro.

Para o cadastro de produtos, primeiramente o usuário deve acessar o menu “Produtos”. Fazendo-se isto, o sistema abre tela que contém a listagem de todos os produtos já

cadastrados. Também é nela que o usuário pode fazer alterações nos itens cadastrados clicando na ação “👤 Mostrar” ou fazer a exclusão dos mesmos clicando na ação “🗑️ Apagar”, conforme mostra a Figura 26.



Figura 26 – Lista de produtos

Para realizar um novo cadastro o usuário deve clicar na opção “Adicionar Produto”. Neste momento o sistema abre a tela para o novo cadastro. Esta tela possui campos obrigatórios, conforme mostra a Figura 27.

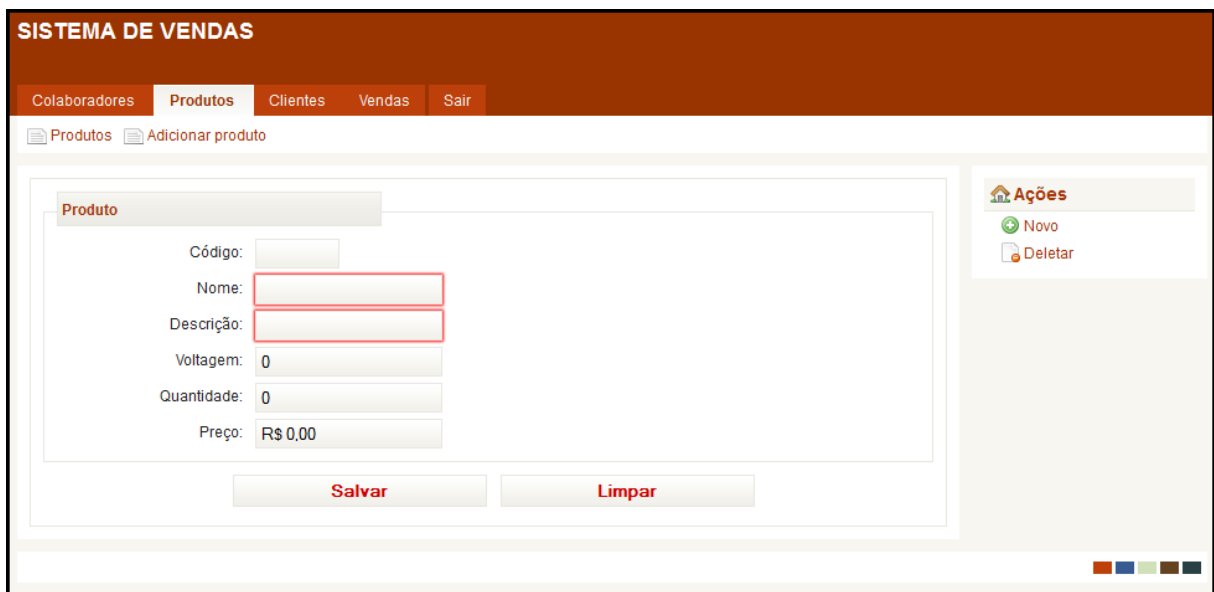


Figura 27 - Novo cadastro de produto

Após o preenchimento dos campos o usuário deve clicar em “Salvar”.

Para realizar o cadastro de clientes, o usuário necessita clicar no menu “Clientes”, para então ter acesso aos clientes que já se encontram no sistema. É por meio deste procedimento que o usuário pode fazer alterações nos itens cadastrados clicando na ação “👤 Mostrar” ou fazer a exclusão dos mesmos clicando na ação “🗑️ Apagar”, conforme mostra a Figura 28.

Código	Nome	CPF/CNPJ	Cidade	CEP	Ação
1	gustavo	7987987987987	Bnu	898908908	
2	Teste	666	Pomba	89107-000	

Figura 28 – Lista de clientes

O sistema abre a tela para o novo cadastro após o usuário clicar em “Adicionar Cliente”. Nesta tela existem campos obrigatórios que devem ser preenchidos, conforme a Figura 29.

Figura 29 - Novo cadastro de cliente

Após o preenchimento dos campos, o usuário deve clicar em “Salvar” para concluir o cadastro.

O administrador tem permissão para efetuar o procedimento de atualização dos boletos em aberto no sistema. Nesta atualização, são verificados através do *web service* quais

boletos foram quitados. Os boletos são gerados *online* no Sistema CoopCob, que é um sistema de cobranças bancárias para cooperativas de crédito.

É por meio deste sistema que atualmente se obtêm informações referentes aos pagamentos que estão aberto e pagamentos quitados para então fazer a atualização das planilhas eletrônicas usadas até então.

Primeiramente o usuário necessita acessar o menu das vendas e em seguida precisa clicar na opção “Atualizar boletos”, como mostra a Figura 30.



Figura 30 - Atualização de boletos

Neste momento o sistema abre a tela em que o usuário deve preencher o período em que deseja fazer a consulta de seus boletos. Essa consulta é feita de acordo com o período de emissão dos boletos gerados. A Figura 31 mostra a tela para a atualização dos boletos.



Figura 31 - Tela de atualização de boletos

No momento em que o usuário preencher os campos de “Data Inicial” e “Data Final” e clicar em “Atualizar”, o *framework* responsável pela comunicação e consumo do *web service* entra em ação.

Com base no arquivo WSDL disponibilizado pelo próprio *web service* e que é apresentado na Figura 32, é possível criar as classes necessárias para a comunicação entre o sistema desenvolvido e o serviço que se deseja acessar. Também é através deste arquivo WSDL que é possível verificar-se os elementos da interface do serviço que está sendo consumido.

```

<?xml:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:tns="http://tempuri.org/" xmlns:s="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://tempuri.org/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
▼<wsdl:types>
  ▼<s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/">
    ▶<s:element name="GerarBoleto">...</s:element>
    ▶<s:element name="GerarBoletoResponse">...</s:element>
    ▶<s:element name="GerarBoletoNN">...</s:element>
    ▶<s:element name="GerarBoletoNNResponse">...</s:element>
    ▶<s:element name="Processos">...</s:element>
    ▶<s:element name="ProcessosResponse">...</s:element>
    ▶<s:element name="Sacados">...</s:element>
    ▶<s:element name="SacadosResponse">...</s:element>
    ▶<s:element name="BoletosDoSacado">...</s:element>
    ▶<s:element name="BoletosDoSacadoResponse">...</s:element>
    ▼<s:element name="BoletosPorPeriodoVencimento">
      ▼<s:complexType>
        ▼<s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="sLogin" type="s:string"/>
          <s:element minOccurs="0" maxOccurs="1" name="sSenha" type="s:string"/>
          <s:element minOccurs="0" maxOccurs="1" name="pDataInicial" type="s:string"/>
          <s:element minOccurs="0" maxOccurs="1" name="pDataFinal" type="s:string"/>
        </s:sequence>
      </s:complexType>
    </s:element>
    ▶<s:element name="BoletosPorPeriodoVencimentoResponse">...</s:element>
    ▶<s:element name="BoletosPorPeriodoEmissao">...</s:element>
    ▶<s:element name="BoletosPorPeriodoEmissaoResponse">...</s:element>
    ▶<s:element name="BoletosPorPeriodoPagamento">...</s:element>
    ▶<s:element name="BoletosPorPeriodoPagamentoResponse">...</s:element>
    ▶<s:element name="GeraImpressao">...</s:element>
    ▶<s:element name="GeraImpressaoResponse">...</s:element>
  </s:schema>
</wsdl:types>
▶<wsdl:message name="GerarBoletoSoapIn">...</wsdl:message>
▶<wsdl:message name="GerarBoletoSoapOut">...</wsdl:message>
▶<wsdl:message name="GerarBoletoNNSoapIn">...</wsdl:message>
▶<wsdl:message name="GerarBoletoNNSoapOut">...</wsdl:message>
▶<wsdl:message name="ProcessosSoapIn">...</wsdl:message>
▶<wsdl:message name="ProcessosSoapOut">...</wsdl:message>
▶<wsdl:message name="SacadosSoapIn">...</wsdl:message>
▶<wsdl:message name="BoletosDoSacadoSoapIn">...</wsdl:message>
▶<wsdl:message name="BoletosDoSacadoSoapOut">...</wsdl:message>
▼<wsdl:message name="BoletosPorPeriodoVencimentoSoapIn">
  <wsdl:part name="parameters" element="tns:BoletosPorPeriodoVencimento"/>
</wsdl:message>
▼<wsdl:message name="BoletosPorPeriodoVencimentoSoapOut">
  <wsdl:part name="parameters" element="tns:BoletosPorPeriodoVencimentoResponse"/>
</wsdl:message>
▶<wsdl:message name="BoletosPorPeriodoEmissaoSoapIn">...</wsdl:message>
▶<wsdl:message name="BoletosPorPeriodoEmissaoSoapOut">...</wsdl:message>
▼<wsdl:message name="BoletosPorPeriodoPagamentoSoapIn">

```

Figura 32 - WSDL do *web service*

As classes geradas através do WSDL são responsáveis por garantir o controle da comunicação do sistema com o *web service*.

A interface *ServicosSoap* define os métodos publicados no *web service* que podem ser utilizados. Na Figura 33 pode-se visualizar a interface *ServicosSoap*.


```

*ServicosSoap.java
/**
 * ServicosSoap.java
 *
 * This file was auto-generated from WSDL
 * by the Apache Axis 1.4.
 */

package org.tempuri;

public interface ServicosSoap extends java.rmi.Remote {

    public java.lang.String gerarBoleto(java.lang.String sLogin,[]

    public java.lang.String gerarBoletoNN(java.lang.String sLogin,[]

    public java.lang.String processos(java.lang.String sLogin,[]

    public java.lang.String sacados(java.lang.String sLogin,[]

    public java.lang.String boletosDoSacado(java.lang.String sLogin,[]

    public java.lang.String boletosPorPeriodoVencimento([]

    //Consulta de boletos é feita através do Período de Emissao
    public java.lang.String boletosPorPeriodoEmissao(java.lang.String sLogin,
        java.lang.String sSenha, java.lang.String pDataInicial,
        java.lang.String pDataFinal) throws java.rmi.RemoteException;

    public java.lang.String boletosPorPeriodoPagamento(java.lang.String sLogin,[]

    public java.lang.String geraImpressao(java.lang.String sLogin,[]
}

```

Figura 33 - Interface “ServicosSoap”

A interface `Servicos` define os métodos do serviço de localização do Web Service, que é implementado pela interface `ServicosLocator`. Na Figura 34 pode-se visualizar a interface `Servicos`.

```

*Servicos.java
/**
 * Servicos.java
 *
 * This file was auto-generated from WSDL
 * by the Apache Axis 1.4.
 */

package org.tempuri;

public interface Servicos extends javax.xml.rpc.Service {

    public java.lang.String getServicosSoapAddress();

    public org.tempuri.ServicosSoap getServicosSoap() throws javax.xml.rpc.ServiceException;

    public org.tempuri.ServicosSoap getServicosSoap(java.net.URL portAddress) throws javax.xml.rpc.ServiceException;
}

```

Figura 34 - Interface "Servicos"

É neste *locator* que fica configurada a URL para o *web service* e que oferece métodos para acessar a classe *stub*, que faz o acesso *proxy* ao *web service*. Na Figura 35 pode-se visualizar a interface `ServicosLocator`.

```

package org.tempuri;

public class ServicosLocator extends org.apache.axis.client.Service implements org.tempuri.Servicos {
    public ServicosLocator() {}
    public ServicosLocator(org.apache.axis.EngineConfiguration config) {}
    public ServicosLocator(java.lang.String wsdlLoc, javax.xml.namespace.QName sName) throws javax.xml.rpc.ServiceException {}

    // Use to get a proxy class for ServicosSoap
    private java.lang.String ServicosSoap_address = "https://www.coopcob.com.br/ws/servicos.asmx";

    public java.lang.String getServicosSoapAddress() {}

    private java.lang.String ServicosSoapWSDDServiceName = "ServicosSoap";

    public java.lang.String getServicosSoapWSDDServiceName() {}

    public void setServicosSoapWSDDServiceName(java.lang.String name) {}

    public org.tempuri.ServicosSoap getServicosSoap() throws javax.xml.rpc.ServiceException {}

    public org.tempuri.ServicosSoap getServicosSoap(java.net.URL portAddress) throws javax.xml.rpc.ServiceException {}

    public void setServicosSoapEndpointAddress(java.lang.String address) {}

    * For the given interface, get the stub implementation.
    public java.rmi.Remote getPort(Class serviceEndpointInterface) throws javax.xml.rpc.ServiceException {}

    * For the given interface, get the stub implementation.
    public java.rmi.Remote getPort(javax.xml.namespace.QName portName, Class serviceEndpointInterface)

    public javax.xml.namespace.QName getServiceName() {}

    private java.util.HashSet ports = null;

    public java.util.Iterator getPorts() {}

    * Set the endpoint address for the specified port name.
    public void setEndpointAddress(java.lang.String portName, java.lang.String address)

```

Figura 35 - Interface "ServicosLocator"

A classe ServicosSoapStub é a raiz, ou seja, a classe que faz a ligação, ou acesso *proxy* entre a aplicação cliente e o *web service*, em outras palavras, faz a comunicação entre o Java e o *web service*. Na Figura 36 pode-se visualizar a interface ServicosSoapStub.

```

public java.lang.String processos(java.lang.String sLogin, java.lang.String sSenha) throws java.rmi.RemoteException {}
public java.lang.String sacados(java.lang.String sLogin, java.lang.String sSenha) throws java.rmi.RemoteException {}
public java.lang.String boletosDoSacado(java.lang.String sLogin, java.lang.String sSenha, java.lang.String pCpfSacado)
public java.lang.String boletosPorPeriodoVencimento(java.lang.String sLogin, java.lang.String sSenha, java.lang.String pDataInici
public java.lang.String boletosPorPeriodoEmissao(java.lang.String sLogin,
    java.lang.String sSenha, java.lang.String pDataInicial, java.lang.String pDataFinal) throws java.rmi.RemoteException {
    if (super.cachedEndpoint == null) {
        throw new org.apache.axis.NoEndpointException();
    }
    org.apache.axis.client.Call _call = createCall();
    _call.setOperation(_operations[6]);
    _call.setUseSOAPAction(true);
    _call.setSOAPActionURI("http://tempuri.org/BoletosPorPeriodoEmissao");
    _call.setEncodingStyle(null);
    _call.setProperty(org.apache.axis.client.Call.SEND_TYPE_ATTR, Boolean.FALSE);
    _call.setProperty(org.apache.axis.AxisEngine.PROP_DOMULTIREFS, Boolean.FALSE);
    _call.setSOAPVersion(org.apache.axis.soap.SOAPConstants.SOAP11_CONSTANTS);
    _call.setOperationName(new javax.xml.namespace.QName("http://tempuri.org/", "BoletosPorPeriodoEmissao"));

    setRequestHeaders(_call);
    setAttachments(_call);
    try {
        java.lang.Object _resp = _call.invoke(new java.lang.Object[] {sLogin, sSenha, pDataInicial, pDataFinal});

        if (_resp instanceof java.rmi.RemoteException) {
            throw (java.rmi.RemoteException) _resp;
        }
        else {
            extractAttachments(_call);
            try {
                return (java.lang.String) _resp;
            } catch (java.lang.Exception _exception) {
                return (java.lang.String) org.apache.axis.utils.JavaUtils.convert(_resp, java.lang.String.class);
            }
        }
    } catch (org.apache.axis.AxisFault axisFaultException) {
        throw axisFaultException;
    }
}

```

Figura 36 – Interface "ServicosSoapStub"

Por fim, a classe UseWS faz a passagem dos parâmetros necessários para receber como retorno os boletos pagos dentro do período desejado e também faz o tratamento do retorno do *web service* para apenas carregar na lista de boletos pagos aqueles que estão cadastrados no sistema. Na Figura 37 pode-se visualizar a interface UseWS.

```

1  /**
2   * recupera a lista de boletos do banco e retorna para ser processada
3   * @param dataInicio
4   * @param dataFim
5   * @return
6   */
7  public static List<Boleto> getListaBoletos(String dataInicio, String dataFim) {
8
9      //Cria uma lista de boleto vazia
10     List<Boleto> listaBoleto = new ArrayList<Boleto>();
11
12     try {
13         //Cria um novo serviço
14         ServicosLocator servico = new ServicosLocator();
15         //Vai no servidor e busca todos os boletos no período.
16         String xmlRetorno = servico.getServicosSoap().boletosPorPeriodoEmissao(login, senha, dataInicio, dataFim);
17         //Transforma o retorno em um input stream
18         ByteArrayInputStream is = new ByteArrayInputStream(xmlRetorno.getBytes());
19
20         DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
21         DocumentBuilder db = dbf.newDocumentBuilder();
22         Document strXML = db.parse(is);
23
24         //Recupera a lista de nodes Boletos
25         NodeList listElementos = strXML.getElementsByTagName("Boletos");
26         //Vare a lista de Boletos recuperando os valores dos nodos filhos
27         for(int i=0; i < listElementos.getLength(); i++){
28             Element ele = (Element) listElementos.item(i);
29             //Vai carregar apenas os boletos que foram pagos
30             if (Double.parseDouble(getTextValue(ele, "VAL_PAGAMENTO")) != 0){
31                 Boleto bol = new Boleto();
32                 bol.setNumeroSacado(getTextValue(ele, "NSACADO"));
33                 bol.setNumeroProcesso(getTextValue(ele, "PROCESSO"));
34                 bol.setDataVencimentoString(getTextValue(ele, "DATA_VENCIMENTO"));
35                 bol.setDataEmissaoString(getTextValue(ele, "DATA_EMISSAO"));
36                 bol.setNumeroParcela(getTextValue(ele, "NUM_PRESTACOES"));
37                 bol.setValorDocumento(Double.parseDouble(getTextValue(ele, "VAL_DOCUMENTO")));
38                 bol.setValorPagamento(Double.parseDouble(getTextValue(ele, "VAL_PAGAMENTO")));
39                 bol.setDataPagamentoString(getTextValue(ele, "DATA_PAGAMENTO"));
40                 bol.setNossoNumero(getTextValue(ele, "NOSSO_NUMERO"));
41                 listaBoleto.add(bol);
42             }
43         }
44         return listaBoleto;
45     } catch (Exception ex) {
46         ex.printStackTrace();
47         return null;
48     }
49 }

```

Figura 37 - Classe "UseWS"

O retorno do *web service* para quando houverem boletos pagos no período e também cadastrados no sistema é demonstrado na Figura 38.

SISTEMA DE VENDAS

Colaboradores Produtos Clientes **Vendas**

Vendas Atualizar boletos

Atualizar boletos - COOPCOB

Filtro da pesquisa

Data Inicial: 01/10/2012

Data Final: 30/12/2012

Atualizar

Lista de boletos atualizados

Nosso número	Data vencimento	Data Emissão	Valor Documento	Valor Pagamento
02655900000051287	22/10/2012	10/10/2012	260.0	260.0

Ações

Figura 38 - Boleto encontrado após consulta ao *web service*

Ao voltar para o cadastro do boleto em questão os campos de data de pagamento e o valor pago aparecem atualizados, conforme Figura 39.

SISTEMA DE VENDAS

Colaboradores Produtos Clientes **Vendas** Sair

Vendas 15/12/2012 Boletos 11

Dados do boleto

Código: 11

Parcela: 1

Nosso número: 02655900000051287

Data emissão: 10/10/2012

Data vencimento: 22/10/2012

Valor do boleto: R\$ 260,00

Dados do pagamento

Valor pago: R\$ 260,00

Data pagamento: 18/10/2012

Salvar **Limpar**

Ações

- Novo
- Deletar

Figura 39 - Dados do boleto atualizados

3.5 RESULTADOS E DISCUSSÃO

O principal objetivo deste trabalho foi desenvolver um sistema de força de vendas para automatizar os processos envolvidos entre a empresa e seus vendedores. Correlacionando a prática atual desta empresa os objetivos específicos do trabalho, nota-se que estes objetivos também foram alcançados e que todos os requisitos propostos foram desenvolvidos.

O aplicativo foi apresentado ao administrador da empresa. Após a sua utilização parcial por um período de aproximadamente uma semana, foi comentado que apesar do sistema ser rápido e funcional, que ele pode ser melhorado em alguns pontos como por exemplo no desenvolvimento de outros métodos de busca a boletos pagos.

Outro ponto comentado foi o da interface, que ainda não está disponível para dispositivos móveis é que seria muito útil uma vez que parte de seus colaboradores se mostra a favor deste tipo de tecnologia.

Baseando-se nos interesses da empresa, decidiu-se por fazer um levantamento relacionando as possíveis melhorias para o sistema, afim de deixa-lo mais robusto e ainda mais funcional. Os itens levantados ainda estão sendo estudados para garantir total satisfação do cliente.

4 CONCLUSÕES

Como conclusão, pode-se analisar que atualmente apesar de existir diversos sistemas que facilitam o dia a dia dos usuários, ainda existem empresas que não tiveram a oportunidade de estarem melhorando os seus processos.

Analisando os objetivos específicos deste trabalho de conclusão de curso, conclui-se que todos foram cumpridos atingindo-se assim o objetivo geral da proposta. Apesar do sistema desenvolvido ainda não estar sendo usado pelos colaboradores, nos testes realizados com o auxílio do administrador da empresa notou-se que o sistema é ágil e eficaz, diminuindo consideravelmente o tempo de execução dos processos atuais da empresa por parte dos futuros colaboradores. Nos testes feitos para verificar a agilidade do sistema com relação a atualização de boletos pagos que hoje é feita por meio de um processo demorado utilizando-se planilhas eletrônicas, o sistema mostrou-se novamente ágil e eficaz atendendo à necessidade do administrador da empresa.

Por meio do contato com o administrador da empresa concluiu-se que apesar dos resultados terem sido bons, a empresa ainda pretende esperar mais algum tempo até que a sua automação nos processos seja feita via software e que neste período poderá ser desenvolvida algumas melhorias para que assim no futuro se tenha um sistema completo de acordo com todas as necessidades da Pereval Comércio de Purificação de Água Ltda.

4.1 EXTENSÕES

Para dar continuidade ao sistema, pode-se desenvolver uma interface *mobile* que contemple todas as plataformas móveis disponíveis no mercado assim como *tablets*.

Pode-se também fazer melhorias na parte da atualização dos boletos e criar um novo módulo específico para tal, para facilitar a visualização dos boletos cadastrados e de seus pagamentos.

Também pode-se disponibilizar os outros métodos de consulta ao *web service* fazer a geração dos boletos dentro do próprio sistema ao invés de gerar eles no ambiente da CoopCob.

REFERÊNCIAS BIBLIOGRÁFICAS

ALBRECHT, Karl. **Revolução nos serviços**. 3. ed. São Paulo: Pioneira, 1992.

APACHE. **The Apache Tomcat 6.0 Servlet**. Los Angeles, [2011]. Disponível em: <<http://tomcat.apache.org/download-60.cgi>>. Acesso em: 12 jul. 2012.

BATISTA, Emerson de Oliveira. **Sistema de Informação: o uso consciente da tecnologia para o gerenciamento**. São Paulo: Editora Saraiva, 2004

CARDOSO, Mário S.; GONÇALVES, Cid F. **CRM em Ambiente e-business**. São Paulo: Atlas, 2001.

CARVALHO, Antonio Vieira de. **Treinamento no Marketing: a força de vendas**. 1.Ed. Rio de Janeiro: Livros Técnicos e Científicos Editora S.A., 1976.

CASA CIVIL. **Lei Nº 8.078: Direitos do consumidor: proteção do consumidor**. Brasil, 1990. Disponível em: <<http://www.aneel.gov.br/cedoc/blei19908078.pdf>>. Acesso em: 29 jul. 2012.

COSTA, Luis Henrique M. K. **Simple Object Access Protocol (SOAP)**: Rio de Janeiro, 2007. Disponível em: < http://www.gta.ufrj.br/grad/07_2/daniel/index.html>. Acesso em: 30 nov. 2012.

GERMANO, Anderson Roberto. **Sistema para consultas e alocação de recursos utilizando Web Services**. Blumenau, 2003. Disponível em: <<http://campeche.inf.furb.br/tccs/2003-I/2003-Iandersonrobertogermanovf.pdf> >. Acesso em: 28 jul. 2012.

KLEIN, Enio. **Os benefícios da Automação da Força de Vendas**. Setembro, 2010. Disponível em: <http://forca-de-vendas.blogspot.com.br/2010_09_01_archive.html> Acesso em: 08 out. 2012.

KREGER, H. **Web services conceptual architecture**. New York, 2001. Disponível em: <<http://www.ibm.com/software/solutions/webservices/pdf/WSCA.pdf>>. Acesso em: 30 ago. 2012.

LAS CASAS, Alexandre Luzzi. **Qualidade total em serviços: conceitos, exercícios, casos práticos**. 4. ed. São Paulo: Atlas, 2004.

MACHADO, Tiago. **Sistema gerencial para automação de força de vendas usando dispositivos móveis baseados em Paml OS**. 2007. 74f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

MEDEIROS, Fábio Tomaz de. **Integração de Sistemas através de Web Services**. São Paulo, 2007. Disponível em: <<http://bibdig.poliseducacional.com.br/document/?view=72>>. Acesso em: 25 set 2012.

MENIN, Juliane. **Gerador de código JSP baseado em projeto de banco de dados MySQL**. 2005. 66f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

MOLLER, Claus. **O lado humano da qualidade: Maximizando a Qualidade de Produtos e Serviços Através do Desenvolvimento das Pessoas**. 1. Ed. São Paulo: Thomson Learning: 1997.

O'BRIEN, James A. **Sistemas de informação e as decisões gerenciais na era da internet**. São Paulo: Editora Saraiva, 2002.

OLIVEIRA, Daniel Quirino. **Livre-se do SQL: uma introdução ao Hibernate**. Outubro, 2003. Disponível em: <<http://www.guj.com.br/articles/125>>. Acesso em: 01 nov. 2012.

PARREIRAS, Fernando. **Serviços web, conceitos e aplicações**. São Paulo, 2011. Disponível em: <<http://www.fernando.parreiras.nom.br/palestras/ws.pdf> >. Acesso em: 14 ago. 2012.

PEPPERS, D.; ROGERS, M. **CRM marketing 1 to 1: um guia executivo para entender e implantar estratégias de customer relationship management**. São Paulo: Peppers and Rogers Group do Brasil, 3.ed. 2004.

PITTELLA, Felipe. **O que é JSP?**. São Paulo, 2006. Disponível em: <<http://www.htmlstaff.org/ver.php?id=1592>>. Acesso em: 21 jul. 2012.

TAVARES, Lúcia. **Setor de serviços acredita em crescimento do faturamento em 2011**. Porto Alegre, 2010. Disponível em: <<http://www.fecomercio-s.org.br/agencia/noticias/Setor-de-servios-acredita-em-crescimentNv8zM-.html>>. Acesso em: 15 set. 2012.

APÊNDICE A – DESCRIÇÃO DOS CASOS DE USO

Nos Quadros de 03 a 09 estão as descrições dos casos de uso do sistema.

Caso de Uso – Login no sistema

Ator: Usuário

Objetivo: Acessar sistema via navegador e informar dados para login e senha armazenados no cadastro de colaborador ou usuário *admin*.

Pré-condições: Sistema deve estar hospedado no servidor *web*; Usuário deve estar cadastrado no banco de dados ou deve ser *admin*.

Pós-condições: Usuário entra conectado ao sistema

Cenário Principal:

1. Usuário preenche seu *login* e sua senha;
 2. Sistema valida os dados de *login* e senha do usuário;
 3. Sistema direciona o Usuário para a página das vendas.
- Cenário Alternativo:** No passo 2, o sistema não reconhece os dados do usuário no sistema
- 2.1 Sistema apresenta mensagem de erro —Usuário ou senha inválida!
 - 2.2 Sistema retorna para página de *login*

Quadro 3 – Login no sistema (UC01)

Caso de Uso – Gerir dados de clientes

Ator: Usuário.

Objetivo: Gerir clientes.

Pré-condições: Usuário estar conectado.

Cenário Principal:

1. Usuário clica no menu “Clientes”
2. O sistema lista os clientes já cadastrados
3. Usuário clica no botão “Adicionar clientes”
4. Usuário preenche os campos necessários
5. Usuário clica em Salvar
6. Sistema valida os campos preenchidos
7. Sistema cadastra o cliente na base de dados

Cenário Alternativo:

No passo 2, se o usuário deseja alterar algum cliente cadastrado:

- 2.1 Usuário encontra o cliente desejado e clica na ação “Mostrar”
- 2.2 Usuário faz as alterações necessárias
- 2.3 Usuário clica em Salvar
- 2.4 Sistema valida os campos preenchidos
- 2.5 Sistema salva as alterações do cliente na base de dados

Cenário Alternativo:

No passo 1, se o usuário deseja excluir algum cliente cadastrado:

- 2.1 Usuário encontra o cliente desejado e clica na ação “Apagar”
- 2.2 Sistema valida a exclusão e apaga o registro em questão.

Quadro 4 – Gerir clientes (UC02)

Caso de Uso – Gerir dados de produtos**Ator:** Usuário.**Objetivo:** Gerir produtos.**Pré-condições:** Usuário estar conectado.**Cenário Principal:**

1. Usuário clica no menu “Produtos”
2. O sistema lista os colaboradores já cadastrados
3. Usuário clica no botão “Adicionar produtos”
4. Usuário preenche os campos necessários
5. Usuário clica em Salvar
6. Sistema valida os campos preenchidos
7. Sistema cadastra o produto na base de dados

Cenário Alternativo:

No passo 2, se o usuário deseja alterar algum produto cadastrado:

- 2.1 Usuário encontra o produto desejado e clica na ação “Mostrar”
- 2.2 Usuário faz as alterações necessárias
- 2.3 Usuário clica em Salvar
- 2.4 Sistema valida os campos preenchidos
- 2.5 Sistema salva as alterações do produto na base de dados

Cenário Alternativo:

No passo 2, se o usuário deseja excluir algum produto cadastrado:

- 2.1 Usuário encontra o produto desejado e clica na ação “Apagar”
- 2.2 Sistema valida a exclusão e apaga o registro em questão.

Quadro 5 – Gerir produtos (UC03)

Caso de Uso – Gerir dados de colaboradores**Ator:** Usuário.**Objetivo:** Gerir colaboradores.**Pré-condições:** Usuário estar conectado.**Cenário Principal:**

1. Usuário clica no menu “Colaborador”
2. O sistema lista os colaboradores já cadastrados
3. Usuário clica no botão “Adicionar colaborador”
4. Usuário preenche os campos necessários
5. Usuário clica em Salvar
6. Sistema valida os campos preenchidos
7. Sistema cadastra o colaborador na base de dados

Cenário Alternativo:

No passo 2, se o usuário deseja alterar algum colaborador cadastrado:

- 2.1 Usuário encontra o colaborador desejado e clica na ação “Mostrar”
- 2.2 Usuário faz as alterações necessárias
- 2.3 Usuário clica em Salvar
- 2.4 Sistema valida os campos preenchidos
- 2.5 Sistema salva as alterações do colaborador na base de dados

Cenário Alternativo:

No passo 1, se o usuário deseja excluir algum colaborador cadastrado:

- 2.1 Usuário encontra o colaborador desejado e clica na ação “Apagar”
- 2.2 Sistema valida a exclusão e apaga o registro em questão.

Quadro 6 – Gerir colaboradores (UC06)

Caso de Uso – Gerir dados das vendas**Ator:** Usuário.**Objetivo:** Gerir vendas.**Pré-condições:** Usuário estar conectado.**Cenário Principal:**

1. Usuário clica no menu “Vendas”
2. O sistema lista os colaboradores já cadastrados
3. Usuário clica no botão “Adicionar venda”
4. Usuário preenche os campos necessários
5. Usuário clica em Salvar
6. Sistema valida os campos preenchidos
7. Sistema cadastra a venda na base de dados

Cenário Alternativo:

No passo 2, se o usuário deseja alterar alguma venda cadastrado:

- 2.1 Usuário encontra a venda desejado e clica na ação “Mostrar”
- 2.2 Usuário faz as alterações necessárias
- 2.3 Usuário clica em Salvar
- 2.4 Sistema valida os campos preenchidos
- 2.5 Sistema salva as alterações da venda na base de dados

Cenário Alternativo:

No passo 2, se o usuário deseja excluir alguma venda cadastrado:

- 2.1 Usuário encontra a venda desejada e clica na ação “Apagar”
- 2.2 Sistema valida a exclusão e apaga o registro em questão.

Quadro 7 – Gerir vendas (UC04)

Caso de Uso – Gerir Boletos**Ator:** Usuário.**Objetivo:** Gerir boletos.**Pré-condições:** Usuário estar conectado.**Cenário Principal:**

1. Usuário clica no botão “Boletos”, dentro da venda
2. O sistema lista os boletos já cadastrados
3. Usuário clica no botão “Adicionar boleto”
4. Usuário preenche os campos necessários
5. Usuário clica em Salvar
6. Sistema valida os campos preenchidos
7. Sistema cadastra o boleto na base de dados

Cenário Alternativo:

No passo 2, se o usuário deseja alterar algum boleto cadastrado:

- 2.1 Usuário encontra o boleto desejado e clica na ação “Mostrar”
- 2.2 Usuário faz as alterações necessárias
- 2.3 Usuário clica em Salvar
- 2.4 Sistema valida os campos preenchidos
- 2.5 Sistema salva as alterações do boleto na base de dados

Cenário Alternativo:

No passo 2, se o usuário deseja excluir algum boleto cadastrado:

- 2.2 Usuário encontra o boleto desejado e clica na ação “Apagar”
- 2.2 Sistema valida a exclusão e apaga o registro em questão.

Quadro 8 – Gerir boletos (UC05)

Caso de Uso – Atualizar boletos**Ator:** Usuário *admin*.**Objetivo:** Atualizar dados de boletos e verificar pagamentos.**Pré-condições:** Usuário *admin* estar conectado.**Cenário Principal:**

1. Usuário acessa o menu “Vendas”
2. Usuário clica na ação “Atualizar boletos”
3. Usuário informa o período a ser verificado
4. Sistema valida os campos preenchidos
5. Usuário clica no botão “Atualizar”
6. Sistema faz a requisição ao *web service*
7. Sistema retorna uma lista com os boletos pagos

Cenário Alternativo:

No passo 7, se não houverem boletos pagos para o período escolhido

- 7.1 Sistema retorna uma lista vazia em tela.

Quadro 9 – Atualização de boletos (UC07)

APÊNDICE B – DICIONÁRIO DE DADOS

Este apêndice apresenta o detalhamento do dicionário de dados, descrevendo o diagrama de modelo entidade relacionamento que foi apresentado da subseção 3.2.4.

Nos Quadros de 10 a 15 estão presentes os dados de todas as tabelas utilizadas pelo sistema. Os campos do tipo “*numeric*” representam valores numéricos. O tipo “*boolean*” representa um campo lógico, podendo ser verdadeiro ou falso. O tipo “*character*” representa uma sequência de letras ou palavras. O tipo “*date*” representa campos que contém datas.

COLABORADOR – Armazena dados dos colaboradores.				
Campo	Descrição	Tipo	Tamanho	Chave primária
id_colaborador	Código do colaborador	Numeric	11	Sim
fl_administrador	Flag administrador	Booelan	1	Não
Nr_agencia	Número da agência	Numeric	11	Não
Ds_bairro	Bairro	Character	255	Não
Ds_banco	Banco	Character	255	Não
Nr_Cep	CEP	Character	255	Não
Ds_cidade	Cidade	Character	255	Não
Cd_conta	Código conta	Numeric	11	Não
Cpf	CPF	Character	15	Não
Dt_nascimento	Data nascimento	Date		Não
Ds_endereco	Endereço	Character	255	Não
Ds_estado	Estado	Character	255	Não
Ds_nome	Nome	Character	255	Não
Senha	Senha	Character	255	Não
Tp_colaborador	Tipo Colaborador	Character	255	Não

Quadro 10 – Tabela Colaborador

VENDA – Armazena dados das vendas.				
Campo	Descrição	Tipo	Tamanho	Chave primária
Id_venda	Código da venda	Numeric	11	Sim
Dt_instalacao	Data da instalação	Date		Não
Dt_venda	Data da venda	Date		Não
Cliente_id_cliente	Código do cliente	Numeric	11	Não
Colaborador_id_colaborador	Cód. do vendedor	Numeric	11	Não

Quadro 11 – Tabela Venda

BOLETOS – Armazena dados dos boletos.				
Campo	Descrição	Tipo	Tamanho	Chave primária
Id_boleto	Código do boleto	Numeric	11	Sim
dataEmissao	Data de emissão	Date		Não
dataPagamento	Data de pagamento	Date		Não
dataVencimento	Data de vencimento	Date		Não
Ds_nosso_numero	Número do cadastro no CoobCop	Character	255	Não
Ds_numero_parcela	Número da parcela	Character	5	Não
Ds_numero_processo	Número do processo	Character	255	Não
Ds_numero_sacado	Número do cliente	Character	255	Não
VI_documento	Valor do boleto	Numeric	5	Não
VI_pagamento	Valor pago	Numeric	5	Não
Venda_id_venda	Código da venda	Numeric	11	Não

Quadro 12 – Tabela Boleto

CLIENTES – Armazena dados dos clientes.				
Campo	Descrição	Tipo	Tamanho	Chave primária
Id_cliente	Código do cliente	Numeric	11	Sim
Ds_bairro	Bairro	Character	255	Não
Nr_cep	Cep	Character	12	Não
Ds_cidade	Cidade	Character	255	Não
Ds_cpf_cnpj	CPF ou CNPJ	Character	20	Não
Ds_endereco	Endereço	Character	255	Não
Ds_estado	Estado	Character	30	Não
Ds_nome	Nome	Character	255	Não
Ds_referencia	Referencia	Character	255	Não
Ds_telefone1	Telefone	Character	12	Não
Ds_telefone2	Telefone	Character	12	Não

Quadro 13 – Tabela Cliente

ITEM – Armazena dados de itens da venda				
Campo	Descrição	Tipo	Tamanho	Chave primária
Id_item	Código do item	Numeric	11	Sim
Nr_quantidade	Quantidade	Numeric	11	Não

Vlr_unitario	Valor unitário	Numeric	5	Não
Produto_id_produto	Código do produto	Numeric	11	Não
Venda_id_venda	Código da venda	Numeric	11	Não

Quadro 14 – Tabela Item

PRODUTO – Armazena dados dos produtos.				
Campo	Descrição	Tipo	Tamanho	Chave primária
Id_produto	Código do produto	Numeric	11	sim
Ds_descricao	Descrição	Character	255	Não
Ds_nome	Nome	Character	255	Não
Vl_sugerido	Valor sugerido	Numeric	5	Não
Qt_estoque	Quantidade em estoque	Numeric	11	Não
Nr_voltagem	Voltagem	Numeric	11	Não

Quadro 15 – Tabela Produto