

UNIVERSIDADE DA REGIÃO DE JOINVILLE – UNIVILLE
DEPARTAMENTO DE INFORMÁTICA

DESENVOLVIMENTO DE UM SISTEMA PARA AGROPECUÁRIA SELVA LTDA ME

GUILHERME POST
PROFESSOR PAULO MARCONDES BOUSFIELD

Joinville - SC

2008

GUILHERME POST

DESENVOLVIMENTO DE UM SISTEMA PARA UMA AGROPECUÁRIA

Trabalho de Conclusão de Estágio apresentado ao curso de Sistemas de Informação da Universidade da Região de Joinville – UNIVILLE – como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação. Orientador Específico: Prof. M.Sc. Paulo Marcondes Bousfield.

Joinville – SC

2008

TERMO DE APROVAÇÃO

O aluno **Guilherme Post**, regularmente matriculado na 4^a série do Bacharelado em Sistemas de Informação da Universidade da Região de Joinville, apresentou e defendeu o presente Trabalho de Conclusão de Estágio, obtendo da Banca Examinadora a média final _____ (_____), tendo sido considerado aprovado.

Joinville, ____ de dezembro de 2008.

*Prof. M.Sc. Paulo
Marcondes Bousfield*

Prof. B

*Prof. Paulo
Marcondes Bousfield
Orientador de classe*

**UNIVERSIDADE DA REGIÃO DE JOINVILLE – UNIVILLE
TRABALHO DE CONCLUSÃO DE ESTÁGIO CURRICULAR SUPERVISIONADO
AVALIAÇÃO DO ESTAGIÁRIO PELA EMPRESA**

Nome do Estagiário: Guilherme Post

QUADRO I

a) AVALIAÇÃO NOS ASPECTOS PROFISSIONAIS	Pontos
1 – QUALIDADE DO TRABALHO – Considerando o possível	
2 – ENGENHOSIDADE – Capacidade de sugerir, projetar, executar modificações ou inovações	
3 – CONHECIMENTO – Demonstrado no desenvolvimento das atividades programadas	
4 – CUMPRIMENTO DAS TAREFAS – Considerar o volume de atividades dentro do padrão razoável	
5 – ESPÍRITO INQUISITIVO – Disposição demonstrada para aprender	
6 – INICIATIVA – No desenvolvimento das atividades	
	SOMA

Pontuação para o Quadro I e II

Sofrível – 1 ponto, Regular – 2 pontos, Bom – 3 pontos, Muito Bom – 4 pontos, Excelente – 5 pontos.

QUADRO II

b) AVALIAÇÃO DOS ASPECTOS HUMANOS	Pontos
1 – ASSIDUIDADE – Cumprimento do horário e ausência de faltas	
2 – DISCIPLINA – Observância das normas internas da Empresa	
3 – SOCIALIZAÇÃO – Facilidade de se integrar com os outros no ambiente de trabalho	
4 – COOPERAÇÃO – Disposição para cooperar com os demais para atender as atividades	
5 – SENSO DE RESPONSABILIDADE – Zelo pelo material, equipamentos e bens da empresa	
	SOMA

c) AVALIAÇÃO FINAL	Pontos
SOMA do Quadro I multiplicada por 7	
SOMA do Quadro II multiplicada por 3	
SOMA TOTAL	

LIMITES PARA CONCEITUAÇÃO
De 57 a 101 – SOFRÍVEL
De 102 a 146 – REGULAR
De 148 a 194 – BOM
De 195 a 240 – MUITO BOM
De 241 a 285 – EXCELENTE

Nome da Empresa: Agropecuária Selva LTDA ME

Representada pelo Supervisor: Emmer Post

**CONCEITO CONFORME
SOMA TOTAL**

**Rubrica do Supervisor da
Empresa**

Local: Joinville

Data:

Carimbo da Empresa

AGRADECIMENTOS

Agradeço a empresa Agropecuária Selva LTDA ME pela oportunidade de desenvolver esse projeto de conclusão de estágio. Agradeço também ao professor Paulo M. Bousfield pela grande ajuda e excelente orientação durante a execução do trabalho.

À minha família: meu pai Emmer, minha mãe Benta, meu irmão Junior, minha irmã Bruna pelo grande apoio desde o início de minha vida acadêmica e a minha namorada Tatieli pelo carinho e compreensão.

Aos meus amigos: pela amizade, força companheirismo e persistência.

Aos meus professores: pelos ensinamentos e por acreditar em cada um de nós.

DEDICATÓRIA

Agradeço a Deus: pela oportunidade de ter esta vida. Dedico a conclusão desse projeto assim como a conclusão de curso à minha família que me ajudaram com mais essa conquista e a minha namorada Tatieli por estar ao meu lado na conclusão de mais uma etapa.

RESUMO

Esse trabalho de conclusão de estágio realizado na Agropecuária Selva, consiste em analisar a necessidade em adquirir um sistema para gerenciar os processos da empresa, projetar um sistema que atenda os requisitos do cliente e implementar o que foi acordado durante as entrevistas feitas com os futuros usuários do sistema. A idéia de se desenvolver um sistema para empresa surgiu por necessidade de gerenciar, pois atualmente a empresa realiza seus processos manualmente. Para atender a esta necessidade, desenvolveu-se um sistema que possibilitará gerenciar os processos de vendas, compras, estoque, contas a pagar e receber. Sendo estas as principais atividades da realizadas pela empresa. Através deste trabalho é possível suprir a necessidade da empresa, alcançando a satisfação do cliente.

LISTA DE FIGURAS

Figura 1: Sistemas de Informação.....	23
Figura 2: Modelo em cascata.....	29
Figura 3: Modelo Espiral.....	29
Figura 4: Modelo Espiral.....	30
Figura 5: Fases e Marco Principais.....	37
Figura 6: Modelo de diagrama de caso de uso	41
Figura 7: Modelo de diagrama de classe.....	42
Figura 8: Modelo de diagrama de seqüência.....	43
Figura 9: Modelo de Diagrama de Colaboração.....	44
Figura 10: Modelo de domínio	46
Figura 11: Diagrama de caso de uso.....	47
Figura 12: Classe de análise <i>boundary</i>	51
Figura 13: Classe de análise <i>control</i>	51
Figura 14: Classe de análise <i>entity</i>	51
Figura 15: Representação simplificada de um sistema de banco de dados.....	53
Figura 16: ANSI/SPARC arquitetura de três esquemas.....	56
Figura 17: Representação gráfica de Entidade.	60
Figura 18: Modelo de domínio	84
Figura 19: Diagrama de Casos de Uso de Contexto	89
Figura 20: Diagrama de Seqüência – Manter cadastro de clientes.....	129
Figura 21: Diagrama de Seqüência – Manter venda.....	130
Figura 22: Diagrama de Seqüência – Manter estoque.	131
Figura 23: Diagrama de Colaboração – Manter cadastro de Fornecedor.	132
Figura 24: Diagrama de Colaboração – Manter cadastro de Funcionário.	132
Figura 25: Diagrama de Colaboração – Manter cadastro de Senha.....	133
Figura 26: Diagrama de Colaboração – Manter cadastro de Condição de Pagamento.....	133
Figura 27: Diagrama de Colaboração – Manter cadastro de Forma de Pagamento.	134
Figura 28: Diagrama de Colaboração – Manter cadastro de Categoria.	134
Figura 29: Diagrama de Colaboração – Manter cadastro de cartão.....	135
Figura 30: Diagrama de Colaboração – Manter cadastro de conta.....	135
Figura 31: Diagrama de Colaboração – Manter cadastro de cheque.	136
Figura 32: Diagrama de Colaboração – Manter cadastro de bairro.....	136
Figura 33: Diagrama de Colaboração – Manter cadastro de contas a pagar.	137
Figura 34: Diagrama de Colaboração – Manter cadastro de contas a receber.	137
Figura 35: Diagrama de Colaboração – Manter cadastro de compras.	138
Figura 36: Diagrama de Colaboração – Manter controle de vendas feitas com cartão.	138
Figura 37: Diagrama de Colaboração – Manter cadastro de nota fiscal de entrada.	139
Figura 38: Diagrama de Colaboração – Manter controle de produtos.....	139
Figura 39: Diagrama de Colaboração – Manter controle estoque.	140
Figura 40: Diagrama de Colaboração – Consultar compras.	140
Figura 41: Diagrama de Colaboração – Consultar clientes.....	141
Figura 42: Diagrama de Colaboração – Consultar fluxo de caixa.....	141
Figura 43: Diagrama de Colaboração – Consultar vendas.....	142
Figura 44: Diagrama de Colaboração – Consultar vendas.....	142
Figura 45: Diagrama de Colaboração – Consultar de contas a pagar.	143
Figura 46: Diagrama de Colaboração – Consultar de contas a receber.	143
Figura 47: Diagrama de Colaboração – Manter cadastro de vendas feitas com cheque.....	144

Figura 48: Diagrama de Colaboração – Gerar relatório de contas a receber	144
Figura 49: Diagrama de Colaboração – Gerar relatório de contas a pagar	145
Figura 50: Diagrama de Colaboração – Gerar relatório de fluxo de caixa	145
Figura 51: Diagrama de Colaboração – Gerar relatório de itens em falta no estoque.....	146
Figura 52: Diagrama de classe detalhado.....	147
Figura 53: Modelagem de dados na 3FN	149
Figura 54: Modelo Entidade Relacionamento.....	150
Figura 55: <i>Layout</i> Login e senha	160
Figura 56: <i>Layout</i> principal	161
Figura 57: <i>Layout</i> cadastro de cliente	162
Figura 58: <i>Layout</i> cadastro de bairro	163
Figura 59: <i>Layout</i> cadastro de vendas.....	164
Figura 60: <i>Layout</i> controle de estoque.....	165
Figura 61: <i>Layout</i> cadastro de funcionário.....	166
Figura 62: <i>Layout</i> cadastro de cadastro de senhas.....	167
Figura 63: <i>Layout</i> cadastro de cadastro de contas a pagar	168
Figura 64: <i>Layout</i> de lançamentos de vendas feitas com cartão	169
Figura 65: <i>Layout</i> cadastro de cadastro de forma de pagamento e condição de pagamento .	170
Figura 66: <i>Layout</i> cadastro de cadastro de nota fiscal de entrada	171
Figura 67: <i>Layout</i> cadastro de cadastro de produtos.....	172
Figura 68: <i>Layout</i> cadastro de cadastro de compra	173
Figura 69: <i>Layout</i> cadastro de cartão	174
Figura 70: <i>Layout</i> cadastro de categoria	175
Figura 71: <i>Layout</i> cadastro de fornecedores	176
Figura 72: <i>Layout</i> consulta de clientes.....	177
Figura 73: <i>Layout</i> consulta de pedidos de compra	178
Figura 74: <i>Layout</i> consulta de contas a pagar	179
Figura 75: <i>Layout</i> consulta de contas a receber.....	180
Figura 76: <i>Layout</i> consulta de vendas.....	181
Figura 77: <i>Layout</i> controle de vendas feitas com cartão.....	182
Figura 78: <i>Layout</i> controle de vendas feitas com cheque	183
Figura 79: <i>Layout</i> de fluxo de caixa	184
Figura 80: <i>Layout</i> que gerar relatório de contas a pagar.....	185
Figura 81: <i>Layout</i> lançamentos de vendas com cheque.....	186
Figura 82: <i>Layout</i> reajuste de preço.....	187
Figura 83: <i>Layout</i> venda resumida	188
Figura 84: <i>Layout</i> do relatório de clientes,.....	189
Figura 85: <i>Layout</i> do relatório de produtos.....	190
Figura 86: <i>Layout</i> do relatório de contas a receber.....	191
Figura 87: <i>Layout</i> do relatório de contas a pagar	192
Figura 88: Código fonte da classe de controle de compras.....	194
Figura 89: Diagrama de Persistência, Cliente Entity	195
Figura 90: Diagrama de Interface	196
Figura 91: Diagrama de Persistência, DAOFactory e ClienteDAO.	197
Figura 92: Classe de testes TesteConsulta.	198

LISTA DE TABELAS

Tabela 1: Cronograma das atividades	20
Tabela 2: Documentação de caso de uso abertura de conta	48
Tabela 3: Dicionário de dados	60
Tabela 4: Regra de negócio – Emitir venda	76
Tabela 5: Regra de negócio – Emitir venda PDV	77
Tabela 6: Regra de negócio – Emitir compra	77
Tabela 7: Regra de negócio – Cadastrar cliente	77
Tabela 8: Regra de negócio – Cadastrar fornecedor	77
Tabela 9: Regra de negócio – Cadastrar nota fiscal de entrada	78
Tabela 10: Regra de negócio – Cadastrar conta	78
Tabela 11: Regra de negócio – Cadastrar funcionário	78
Tabela 12: Regra de negócio – Clientes inadimplentes	78
Tabela 13: Regra de negócio – Pedidos retidos	79
Tabela 14: Regra de negócio – Clientes não cadastrados	79
Tabela 15: Regra de negócio – Atendente não cadastrado	79
Tabela 16: Regra de negócio – Recebimento de nota fiscal em desacordo	79
Tabela 17: Regra de negócio – Cadastro de produtos	79
Tabela 18: Regra de negócio – Cadastro de categoria	79
Tabela 19: Regra de negócio – Cadastrar de pagamentos	80
Tabela 20: Regra de negócio – Cadastro de forma de pagamento	80
Tabela 21: Regra de negócio – Cadastrar condição de pagamento	80
Tabela 22: Regra de negócio – Cadastro de vendas com cartão	80
Tabela 23: Regra de negócio – Gerar relatórios contas a pagar	80
Tabela 24: Regra de negócio – Cadastro de vendas com cheque	81
Tabela 25: Regra de negócio – Cadastro de senhas	81
Tabela 26: Regra de negócio – Gerar relatório de produtos em falta no estoque	81
Tabela 27: Regra de negócio – Gerar relatórios contas a receber	81
Tabela 28: Regra de negócio – Gerar relatórios fluxo de caixa	81
Tabela 29: Requisitos do Sistema	82
Tabela 30: Requisitos não Funcionais	83
Tabela 31: Definição dos atores do sistema	85
Tabela 32: Definição dos casos de uso	85
Tabela 33: Priorização de Casos de Uso	87
Tabela 34: Caso de Uso – Efetuar <i>login</i>	90
Tabela 35: Caso de Uso – Manter cadastro de funcionário	90
Tabela 36: Caso de Uso – Manter cadastro de senhas	91
Tabela 37: Caso de Uso – Consultar produtos	91
Tabela 38: Caso de Uso – Manter cadastro de compras	92
Tabela 39: Caso de Uso – Consultar pedido de compra	93
Tabela 40: Caso de Uso – Manter Cadastro de Nota Fiscal de Entrada	94
Tabela 41: Caso de Uso – Manter cadastro de clientes	95
Tabela 42: Caso de Uso – Manter cadastro de fornecedores	95
Tabela 43: Caso de Uso – Manter pedido de venda	96
Tabela 44: Caso de Uso – Consultar venda	97
Tabela 45: Caso de Uso – Consultar clientes	98
Tabela 46: Caso de Uso – Cadastrar condição de pagamento	99
Tabela 47: Caso de Uso – Manter cadastro de forma de pagamento	99

Tabela 48: Caso de Uso – Manter cadastro de produto	100
Tabela 49: Caso de Uso – Manter cadastro de contas a pagar	100
Tabela 50: Caso de Uso – Consultar recebimentos	101
Tabela 51: Caso de Uso – Consultar contas a pagar.....	101
Tabela 52: Caso de Uso – Manter cadastro de categoria	102
Tabela 53: Caso de Uso – Manter cadastro de bairro	102
Tabela 54: Caso de Uso – Manter cadastro de vendas com cheques.....	103
Tabela 55: Caso de Uso – Manter cadastro de vendas com cartão.....	103
Tabela 56: Caso de Uso – Manter cadastro de contas.	104
Tabela 57: Caso de Uso – Manter cadastro de cartão.....	104
Tabela 58: Caso de Uso – Manter cadastro de cartão.....	105
Tabela 59: Caso de Uso – Controlar vendas feitas com cheques.	105
Tabela 60: Caso de Uso – Consultar fluxo de caixa.....	106
Tabela 61: Caso de Uso – Controlar vendas feitas com cartão.....	106
Tabela 62: Caso de Uso – Gerar relatório de contas a receber.....	107
Tabela 63: Caso de Uso – Gerar relatório de contas a pagar	107
Tabela 64: Caso de Uso – Gerar relatório de produtos em falta no estoque	108
Tabela 65: Caso de Uso – Gerar relatório de fluxo de caixa.....	108
Tabela 66: Analisar Casos de Uso – Efetuar login.....	109
Tabela 67: Analisar Casos de Uso – Manter cadastro de funcionário.....	110
Tabela 68: Analisar Casos de Uso – Manter cadastro de cliente.....	110
Tabela 69: Analisar Casos de Uso – Manter cadastro de produto.....	111
Tabela 70: Analisar Casos de Uso – Manter cadastro de cartão.	111
Tabela 71: Analisar Casos de Uso – Manter cadastro de compra.	112
Tabela 72: Analisar Casos de Uso – Manter cadastro de condição de pagamento.	112
Tabela 73: Analisar Casos de Uso – Manter cadastro de forma de pagamento.	113
Tabela 74: Analisar Casos de Uso – Manter cadastro de bairro.....	113
Tabela 75: Analisar Casos de Uso – Manter cadastro categoria.	114
Tabela 76: Analisar Casos de Uso – Manter cadastro de contas a pagar.....	114
Tabela 77: Analisar Casos de Uso – Manter cadastro de contas a receber.....	115
Tabela 78: Analisar Casos de Uso – Manter controle de estoque.	115
Tabela 79: Analisar Casos de Uso – Manter cadastro de senhas.....	116
Tabela 80: Analisar Casos de Uso – Manter cadastro de conta.	116
Tabela 81: Analisar Casos de Uso – Manter venda com cartão.	117
Tabela 82: Analisar Casos de Uso – Consultar fluxo de caixa.....	117
Tabela 83: Analisar Casos de Uso – Manter venda com cheque.....	118
Tabela 84: Analisar Casos de Uso – Controlar vendas com cheques.....	118
Tabela 85: Analisar Casos de Uso – Consultar venda.	119
Tabela 86: Analisar Casos de Uso – Consultar contas a pagar.	119
Tabela 87: Analisar Casos de Uso – Consultar contas a receber.	120
Tabela 88: Analisar Casos de Uso – Consultar compras.	120
Tabela 89: Analisar Casos de Uso – Consultar cliente.	121
Tabela 90: Analisar Casos de Uso – Consultar produto.	121
Tabela 91: Analisar Casos de Uso – Controlar vendas com cartão.	122
Tabela 92: Analisar Casos de Uso – Manter cadastro de nota fiscal de entrada.	122
Tabela 93: Analisar Casos de Uso – Gerar relatório de contas a receber.	123
Tabela 94: Analisar Casos de Uso – Gerar relatório de contas a pagar.....	124
Tabela 95: Analisar Casos de Uso – Gerar relatório de produtos em falta no estoque.	125
Tabela 96: Analisar Casos de Uso – Gerar relatório de fluxo de caixa.	126
Tabela 97: Analisar Casos de Uso – Manter cadastro de fornecedor.	127

Tabela 98: Analisar Casos de Uso – Manter cadastro de venda.....	128
Tabela 99: Dicionário de dados da tabela cliente	151
Tabela 100: Dicionário de dados da tabela de cheques	152
Tabela 101: Dicionário de dados da tabela de caixa.....	152
Tabela 102: Dicionário de dados da tabela de produtos	153
Tabela 103: Dicionário de dados da tabela de cartão	153
Tabela 104: Dicionário de dados da tabela de funcionários.....	154
Tabela 105: Dicionário de dados da tabela de forma de pagamento	154
Tabela 106: Dicionário de dados da tabela de fornecedores.....	155
Tabela 107: Dicionário de dados da tabela de contas a pagar.....	155
Tabela 108: Dicionário de dados da tabela de vendas	156
Tabela 109: Dicionário de dados da tabela de item da venda	156
Tabela 110: Dicionário de dados da tabela de conta a receber	157
Tabela 111: Dicionário de dados da tabela de compras.....	157
Tabela 112: Dicionário de dados da tabela de itens da compra	158
Tabela 113: Dicionário de dados da tabela de condição de pagamento	158
Tabela 114: Dicionário de dados da tabela de vendas com cartão	158
Tabela 115: Dicionário de dados da tabela de contas	159
Tabela 116: Dicionário de dados da tabela de produtos e fornecedores.....	159
Tabela 117: Dicionário de dados da tabela de senhas	159

SUMÁRIO

RESUMO	7
LISTA DE FIGURAS	8
LISTA DE TABELAS	10
SUMÁRIO	13
INTRODUÇÃO	15
1. DEFINIÇÃO DO PROJETO	16
1.1. Dados de Identificação do Aluno	16
1.2. Dados de Identificação da Empresa	16
1.3. Dados dos Responsáveis pelo Estágio	16
1.4. Empresa	16
1.5. Tema	17
1.6. Problema	17
1.7. Objetivos	18
1.7.1. Objetivo Geral	18
1.7.2. Objetivos Específicos	18
1.7.3. Justificativa	18
1.7.4. Metodologia	19
1.7.5. Cronograma	20
2. FUNDAMENTAÇÃO TEÓRICA	21
2.1. Sistema	21
2.2. Sistemas de Informação	22
2.3. Classificação dos Sistemas de Informação	24
2.4. Engenharia de <i>Software</i>	25
2.5. Ciclo de Vida dos Sistemas de Informação	27
2.6. Metodologia de desenvolvimento de sistemas de informação	31
2.7. Processo Unificado (<i>Unified Process</i>)	32
2.7.1. Princípios Chaves do Processo Unificado	33
2.7.2. As Fases do Processo Unificado	35
2.8. <i>Rational Unified Process (RUP)</i>	37
2.9. <i>Unified Modeling Language (UML)</i>	38
2.9.1. Diagrama de Casos de Uso	40
2.9.2. Diagrama de classe	41
2.9.3. Diagrama de Seqüência	42
2.9.4. Diagrama de Colaboração	43
2.10. Levantamento de Requisitos	44
2.10.1. Modelo de Domínio	45
2.10.2. Encontrar Atores e Caso de Uso	46
2.10.3. Priorizar Caso de Uso	47
2.10.4. Detalhamento do Caso de Uso	48
2.10.5. Protótipo de <i>Interface</i> do Usuário	49
2.10.6. Estrutura do Modelo de Caso de Uso	49
2.11. Análise dos requisitos	50
2.11.1. Analisar Arquitetura	50
2.11.2. Analisar Caso de uso	51
2.12. Projeto	52
2.12.1. Banco de Dados	52
2.12.2. Sistema Gerenciador de Banco de Dados	53

2.12.3. Banco de Dados Relacional	56
2.12.4. Modelagem de Dados	57
2.12.5. MER – Modelo Entidade Relacionamento	59
2.12.6. Dicionário de Dados	60
2.12.7. SQL (Structured Query Languangue)	60
2.13. Implementação	62
2.13.1. Linguagens de Programação	62
2.13.1.1. Linguagem Java	65
2.13.2. Integração	66
2.14. Testes	67
2.15. Implantação	70
2.16. Manutenção de Sistemas	70
2.17. Processos Operacionais	71
2.17.1. Vendas	71
2.17.2. Contas a Receber	72
2.17.3. Contas a Pagar	73
2.17.4. Compras	73
2.17.5. Controle de Estoque	74
3. DESCRIÇÃO PRÁTICA	75
3.1. Local de Desenvolvimento do Sistema	75
3.2. Sistema atual da empresa	75
3.3. Regras de Negócios	76
3.4. Levantamento de Requisitos	82
3.4.1. Modelo de Domínio do Sistema	83
3.4.2. Encontrar atores	84
3.4.3. Encontrar caso de uso	85
3.4.4. Prioridade dos Casos de Uso	87
3.4.5. Definir Caso de Uso	88
3.4.6. Detalhar Caso de Uso	90
3.5. Análise de Requisitos	108
3.5.1. Analisar Arquitetura	109
3.5.2. Analisar Caso de Uso	109
3.5.3. Diagrama de Seqüência	128
3.5.4. Diagrama de Colaboração	131
3.5.5. Analisar Classe	146
3.6. Projeto (<i>Design</i>)	147
3.6.1. Projetar Banco de Dados	148
3.6.2. Modelagem de dados (Normalização – 3FN)	148
3.6.3. MER (Modelo Entidade Relacionamento)	150
3.6.4. Dicionário de dados	150
3.7. Projetar Interface	159
3.8. Implementação	192
3.9. Testes	197
3.10. Implantação	199
CONSIDERAÇÕES FINAIS	201
REFERÊNCIAS	202

INTRODUÇÃO

O presente trabalho tem como objetivo descrever o estágio realizado na empresa Agropecuária Selva LTDA ME, localizada na cidade de Joinville, atuando no segmento de comercialização de rações para animais de pequeno e médio porte. Neste trabalho são apresentados os problemas e a solução, que surgiu através do desenvolvimento de um sistema para gerenciar os processos que hoje são feitos manualmente pelas pessoas que trabalham na empresa.

Para alcançar o êxito no desenvolvimento do sistema, o projeto foi conduzido de uma forma metodológica desde seu início, utilizando como metodologia o Processo Unificado e como Linguagem para Modelagem dos dados a UML. O relatório a seguir é dividido em três capítulos, a definição do Projeto, Fundamentação Teórica e a Descrição Prática.

A Definição do Projeto apresenta o entendimento do negócio, a situação atual da empresa, qual o problema encontrado e a proposta oferecida para empresa. De uma forma clara, através do cronograma é possível visualizar as etapas e o tempo que levará para desenvolver o sistema.

A Fundamentação Teórica foi descrita através dos conceitos encontrados nos livros pesquisados no decorrer do estágio. O embasamento teórico foi fundamental para esclarecimentos e enriquecimento do conhecimento.

Com o estudo de técnicas e conceitos importantes de desenvolvimento, no terceiro capítulo é feita uma Descrição do levantamento de requisitos, análise, projeto do sistema, implementação, testes e implantação. Neste capítulo ocorre uma união, onde é apresentada a teoria na prática, para alcançar os objetivos, de atender os requisitos do cliente, desenvolver um sistema com qualidade, confiável e seguro.

1. DEFINIÇÃO DO PROJETO

1.1. Dados de Identificação do Aluno

Nome: Guilherme Post.

Curso: Bacharelado em Sistemas de Informação.

Endereço: Rua Das Telefonistas, nº 104- Petrópolis - Joinville – SC.

1.2. Dados de Identificação da Empresa

Denominação: Agropecuária Selva LTDA ME.

Ramo de Atividade: Comércio de Rações (Pet Shopping).

Endereço: Avenida Paulo Schroede, 1995 – Petrópolis - Joinville - SC.

Fone: (47) 3436-8850 - e-mail: agro.selva@ig.com.br

1.3. Dados dos Responsáveis pelo Estágio

Orientador de Classe: Prof. Paulo Marcondes Bousfield.

Orientador Específico: Prof. Paulo Marcondes Bousfield.

Supervisor no Campo de Estágio: Emmer Post.

1.4. Empresa

A Agropecuária Selva foi fundada em junho de 1994 na cidade de Joinville em Santa Catarina pelo Sr. Celso Coelho. A empresa começou suas atividades comercializando rações, ferragens, medicamentos para atender os clientes da comunidade.

Em junho de 1996 o Sr. Emmer Post fez a compra do estabelecimento, incluindo nome da empresa, ponto comercial e o estoque da loja. A partir deste momento a empresa continua suas atividades, mas sob nova direção. Como o novo proprietário havia adquirido experiências de outros trabalhos começou a fazer as mudanças. Aumentando a variedade de tipos de rações para animais de pequeno, médio e grande porte. Investindo também na variedade de ferragens, materiais de pesca, medicamentos para todo o tipo de animais, materiais elétricos, hidráulicos, árvores frutíferas, colocando ainda animais como cachorro,

gato, pássaros para comercialização. Fazendo assim com que a empresa começasse a crescer no mercado e se tornar uma tradição na zona sul de Joinville.

1.5. Tema

Construção de um sistema para gerenciar os processos de uma agropecuária.

1.4 Assunto

A empresa tem a necessidade de controlar com uma maior precisão os seus processos, visando organizar os dados, agilizar os processos, otimizando as operações de rotina da empresa.

Com este sistema o trabalho da empresa será bem mais rápido e eficiente, diminuindo a possibilidade de erros na tarefa repetitiva de vendas e compras.

1.6. Problema

A empresa Agropecuária Selva atual no mercado de Pet Shop há 11 anos, suas principais características são oferecer ao seu cliente um ótimo atendimento, produtos e preços competitivos.

Com o crescimento contínuo da empresa, a necessidade de se obter um sistema que controle os processos de vendas, compras e controle de estoque, cresce a cada dia mais. Por que os processos hoje dentro da empresa são realizados manualmente, ou seja, são feitas anotações em agendas informando a data de vencimento de um boleto, ou o dia que se deve fazer um depósito para cobrir um cheque, o que se vende a vista não é anotado, as vendas feitas a prazo são anotadas em pequenos papéis.

Com a falta de informatização destes processos há inexistência do gerenciamento ficando impossível obter informações sobre a situação atual do estoque, qual o valor que se esperar receber ou pagar, o quanto se tem em caixa. Não existe o controle dos cadastros de produtos, clientes, vendas, compras, da nota fiscal de entrada, nota fiscal de saída, devido a este problema não existe a possibilidade de gerarem-se relatórios.

Com a ausência de um sistema para gerenciar processos, ocorrem problemas com a falta de controle, ficando impossível gerar relatórios e causando transtornos devido à falta de

uma ferramenta apropriada de gerenciamento, ficando ainda impossível de realizar um planejamento e de se criar estratégias devido não possuir dados concretos.

1.7. Objetivos

1.7.1. Objetivo Geral

Desenvolver um sistema de informação para as áreas de venda, controle de estoque e financeiro, fornecendo ao usuário dados de cada processo.

1.7.2. Objetivos Específicos

- a) Identificar as necessidades e analisar os processos desenvolvidos atualmente pela empresa;
- b) Desenvolver módulos para facilitar a divisão de processos da empresa para obter maior precisão nas informações;
- c) Desenvolver relatórios para permitir ao usuário realizar as tomadas de decisões;
- d) Criar um banco de dados para manter o cadastro de clientes, fornecedores, usuários, pedidos de venda, pedidos de compra, produtos, contas a pagar e contas a receber;
- e) Realizar testes no sistema;
- f) Implantar o sistema.

1.7.3. Justificativa

Atualmente, os clientes não querem somente cordialidade, mas sim ter um atendimento rápido e de qualidade. As empresas que alcançam estes objetivos conseguem fazer com que os clientes se sintam a vontade para retornar a empresa para comprar mais e deixando uma boa impressão no mercado, para assim tornar-se mais competitiva. Para tanto, os clientes, ao observarem uma empresa que possui um sistema para gerenciar processos, desde controlar estoque até manter as vendas, esta empresa estará alcançando mais clientes para o seu

estabelecimento, informando a demais membros da sociedade que nesta empresa o cliente tem segurança, conforto, agilidade e qualidade no atendimento.

Segundo (Fernandes, 1995) "...as empresas precisam obter um maior controle de seus processos, para que possam proporcionar ao cliente um ambiente confiável e de satisfação, tendo em vista esta necessidade o sistema possibilitará gerenciar os processos da empresa".

Como já é de conhecimento da empresa, existe uma grande quantidade de sistemas para gerenciamento de processos para micro empresas, o diferencial deste sistema, é que será desenvolvido de acordo com a real necessidade do cliente. Com este sistema será possível automatizar os processos, que atualmente na empresa são feitos manualmente, este sistema irá garantir segurança, qualidade e fácil utilização pelo usuário, proporcionando a empresa uma excelente escolha na aquisição deste sistema.

Este projeto de Sistemas de Informação vem ao encontro com a necessidade da empresa, que é de aumentar o controle e formalizar os processos da empresa, trazendo benefícios à empresa, gerenciando e organizando melhor seus processos.

1.7.4. Metodologia

A pesquisa bibliográfica será através das entrevistas com os usuários e a pesquisa de campo serão os tipos de pesquisas utilizados neste trabalho.

Deverá ser utilizada para o desenvolvimento deste sistema, a análise orientada ao objeto e a modelagem será realizada com os padrões UML 2 (*Unified Modeling Language* - padronização da linguagem de desenvolvimento orientado a objetos), fazendo uso de seus diagramas.

A implementação deste sistema será feita através da linguagem Java utilizando a plataforma Eclipse 3.2 por ser uma linguagem que oferecer maiores recursos, como banco de dados será utilizado o SQL Server 2000 (sistema gerenciador de Banco de dados relacional criado pela Microsoft).

1.7.5. Cronograma

Tabela 1: Cronograma das atividades.

ATIVIDADES	01	02	03	04	05	06	07	08	09	10	11	12
Levantamento das informações para a elaboração do projeto	X	X	X									
Estudo do negocio e pesquisa da melhor linguagem a usar				X								
Preparação do material, fundamentação teórica, pesquisa em livros e artigos.					X	X	X	X				
Aprendizado da linguagem e da técnica de desenvolvimento web.							X	X	X			
Desenvolvimento do projeto, layout do portal e funções existentes.									X	X	X	
Conclusões e considerações finais do projeto												X

Fonte: O Autor.

Legenda:

- | | | |
|-----|----|---|
| 1- | 01 | - indica o mês de Janeiro |
| 2- | 02 | - indica o mês de Fevereiro |
| 3- | 03 | - indica o mês de Março |
| 4- | 04 | - indica o mês de Abril |
| 5- | 05 | - indica o mês de Maio |
| 6- | 06 | - indica o mês de Junho |
| 7- | 07 | - indica o mês de Julho |
| 8- | 08 | - indica o mês de Agosto |
| 9- | 09 | - indica o mês de Setembro |
| 10- | 10 | - indica o mês de Outubro |
| 11- | 11 | - indica o mês de Novembro |
| 12- | 12 | - indica o mês de Dezembro |
| 13- | X | - indica o período em que será realizada a atividade. |
| 14- | | - Indica o período em que não será realizada a atividade. |

2. FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados todos os conceitos relacionados ao desenvolvimento de sistemas que posteriormente será utilizado como base para a elaboração da parte prática do projeto.

2.1. Sistema

Vários conceitos são utilizados sobre sistemas. Sistema é todo procedimento ou rotina criada ou não pelo homem com processos diferentes, mas com um único objetivo.

Conforme REZENDE (2005, pg. 13) sistemas é:

“um conjunto de partes que interagem entre si, integrando-se para atingir um objetivo ou resultado; partes interagentes e interdependentes que formam um todo unitário com determinados objetivos e efetuam determinadas funções; em informática, é o conjunto de *software*, hardware e recursos humanos; componentes da tecnologia da informação e seus recursos integrados; empresa ou organização e seus vários subsistemas”.

Ao analisar o corpo humano é possível identificar vários tipos de sistemas, tais como: sistema circular, sistema respiratório, sistema reprodutor assim como muitos outros, que buscam manter o equilíbrio do corpo humano mantendo o ser humano vivo. Assim que um deles falhar todos os sistemas que estão ligados diretamente ou indiretamente será afetado.

A idéia principal de um sistema é reunir um número de determinados processos, atividades e funções que efetuaram diversos tipos de trabalhos, mas em busca de um objetivo em comum.

ALBERTÃO (2001, pg. 74) “conceitua-se como sendo um conjunto de elementos ou de componentes que interagem em busca de um só objetivo, uma necessidade ou em função de uma dada finalidade, visando atender à missão de uma empresa”.

O conceito de sistema em informática é analisado como um conjunto de componentes como *software*, *hardware* e recursos humanos.

Analizando estes conceitos de sistema citado entende-se que o elemento principal do sistema é a informação. Para haver um objetivo ou resultado é preciso saber que tipo de informação a organização vai precisar, para assim saber de que forma armazenar, tratar e fornecer a informação de tal modo que possa apoiar os processos da organização. As atividades são diversas, mas o objetivo é tratar a informação que será gerada pelos processos e

atividades para posteriormente apresenta – lá ao usuário da forma mais clara de se entender ou interpretar.

2.2. Sistemas de Informação

A empresa e seu contexto, por si só, já constituem um sistema e, em consequência, um Sistema de Informação REZENDE (1999). Que hoje se percebe que é a informação é a arma principal para uma organização se manter competitiva no mercado, os sistemas de informação permitem trabalhar a informação e apresentá-la ao usuário de uma forma mais clara e objetiva.

LAUDON & LAUDON (1999, pg. 4) define que:

Sistema de informação pode ser um conjunto de componentes interrelacionados que coleta (ou recupera), processa, armazena e distribui informações para dar suporte à tomada de decisão e ao controle da organização.

Toda organização gera informação de diversas formas, as empresas que procuram trabalhar as informações dentro ou fora de sua própria organização, com a informação conseguem descobrir tendências, gargalos e oportunidades para se tornarem mais competitivas. Sistemas de informação contêm informações sobre pessoas, lugares e coisas de interesse, no ambiente ao redor da organização e dentro da própria organização.

Sistemas de informação é a forma a qual é manipulada a informação dentro dos diversos tipos de sistemas. Ao analisar um sistema computacional, que através da interação humana, usa de recursos de hardware e *software* para armazenar e processar informações. Para REZENDE (2005, pg. 18) a informação é o principal meio de sobrevivência das organizações por esta razão afirma que:

A informação é um recurso efetivo e inexorável para as organizações, principalmente quando planejada e disponibilizada de forma personalizada, com qualidade inquestionável e preferencialmente antecipada para facilitar as decisões.

Os sistemas de informação podem ser divididos em três atividades básicas: entrada, processamento e saída. A entrada é onde ocorre à captação dos dados, podendo ser dentro da organização ou do ambiente externo, muitas organizações antes de lançar um produto realizam pesquisas no mercado para identificar as necessidades dos clientes.

O processamento é onde ocorre à mineração desses dados para serem devolvida de uma forma mais útil e apropriada, neste caso será filtrado e desenvolvido um produto que esteja dentro dos requisitos do cliente.

A saída ocorre após o processamento onde serão levadas as pessoas a que interessarem ou atividades que a usarão, neste momento ocorrerá à divulgação do produto ou a entrega do mesmo aos clientes que o solicitaram.

O autor REZENDE (1999) define alerta que todo sistema, usando ou não recursos de Tecnologia da informação, que manipula e gera informação pode ser genericamente considerado Sistema de Informação, identifica-se desta forma que existe uma gama muito grande de sistemas de informação que estão no dia a dia das pessoas.

Após analisar estes conceitos percebe-se a importância do uso de sistemas de informação dentro das organizações. Identifica-se que os sistemas de informação se tornaram um aliado das organizações, podendo trazer muitos recursos que possibilitarão as organizações se tornarem mais competitivas. Auxiliando nas tomadas de decisões, podendo enxergar tendências, ajustar gargalos e assim fortalecendo o crescimento da organização.

Com o crescimento contínuo das organizações, percebe-se que quem detém a informação e a utiliza da forma correta conquista espaço para sobreviver no mercado. A captura da informação em muitas organizações se torna um trabalho difícil que exigirá um esforço de todos os envolvidos, mas que pode se conquistada com o auxílio de um sistema de informação que possibilite armazenar, processar e mostrar a informação mais clara e objetiva.

Na figura 1 á uma representação das atividades dos sistemas de informação.

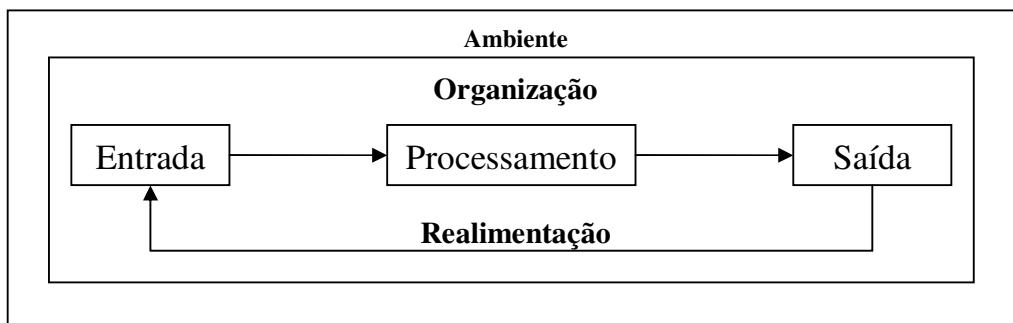


Figura 1: Sistemas de Informação.
Fonte: Laudon & Laudon (1999, pg. 4).

2.3. Classificação dos Sistemas de Informação

Os sistemas de informação que encontramos são vários, desde o sistema de controle aéreo utilizado para navegação de aeronaves a um sistema de gerenciamento das organizações. A grande maioria desses sistemas está ao nosso redor e com certeza fazemos parte deste sistema e da sua rotina.

Assim consegue-se identificar dois tipos de sistemas, fechado e aberto. Os sistemas fechados são aqueles onde só à interação dos componentes entre si, esses sistemas não influenciam nem causam influências ao meio externo. Conforme REZENDE (1999, pg. 132) “sistema fechado segue a mesma idéia, é isolado, hermético, independente e sem abordagem sistêmica, sem receber influência qualquer ao mesmo e também sem permitir influenciar o meio ambiente externo”.

Os sistemas abertos agem de forma contraria, interagem com o meio externo podendo assim, sofrer influências de outros sistemas ou subsistemas. REZENDE (1999, pg. 132) afirma que “sistema aberto tem relações de troca e interdependência dos demais sistemas a sua volta, com abordagem sistêmica, possibilitando receber influências e influenciar os outros sistemas externos a ele”.

Alguns tipos de sistemas são destacados abaixo:

- Sistemas de informação baseados em computador (SIBC): É o procedimento de captura da informação, processamento e exibição da informação com mais clareza. Segundo ALBERTÃO (2001, pg. 77) “tem como função coletar, manipular e processar dados, transformando em informação e cujos componentes são: hardware, software, banco de dados, pessoas, procedimentos e telecomunicações”.
- Sistema de processamento de transações (SPT): Surgiu com a necessidade das organizações de redução de custos, utilizado em negócios para gerenciar procedimentos repetitivos e rotineiros. Facilitando processos, auxiliando nas atividades, fazendo com que a organização conseguisse baixar custos. ALBERTÃO (2001, pg. 79) relata que o primeiro sistema baseado nestas condições foi o sistema de folha de pagamento, seguido pelos sistemas de faturamento e controle de estoque.
- Sistemas de informação gerenciais (SIG): Surgiu da evolução do SPT, mas com o intuito de fornecer a informação ao usuário de uma forma melhor que auxiliará em muito os executivos nas tomadas de decisões. REZENDE (2000, pg. 135) relata que os

sistemas de informação gerencial “contemplam o processamento de grupos de dados das operações operacionais e transações gerenciais, transformando-os em informações estratégicas”. Sendo possível analisar a informação através de relatórios que possibilitam uma melhor visualização das situações que estão ocorrendo dentro da organização. Como exemplo a emissão de relatórios que apontam quais itens estão abaixo do estoque de segurança. Podendo assim preparar um pedido de compra com antecedência e tempo para analisar os preços de fornecedores e efetuar a melhor compra.

- Sistemas de apoio à decisão (SAD): conhecidos também por alguns autores como sistemas de Apoio às operações Empresariais. O SAD foi desenvolvido baseado no SIG, mas com o objetivo de atender problemas mais complexos que não poderiam ser solucionados pelo SIG. ALBERTÃO (2001, pg. 99) cita como exemplo “que o SAD poderia ser utilizado para determinar qual o melhor lugar para realizar perfurações de poços de petróleo dentro dos parâmetros fornecidos”. Tornando assim o SAD como um *software* que ajuda na tomada de decisão. Tem a características de controlar procedimento e rotinas diárias, tornando possível armazenar informações que futuramente serão analisadas e utilizadas por equipes que compõe a organização auxiliando na tomada de decisão.

Sistemas baseados em conhecimento: É um processo de captura da informação que pode ser compreendida de acordo com a capacidade de cada indivíduo envolvido na organização. RESENDE (2003) comenta que o sistema de conhecimento manipula e gera conhecimentos a partir das “bases de conhecimento”. Essa base é onde é armazenada o conhecimento expresso em dados não triviais, imagens, sons, raciocínios elaborados. Para uma organização obter vantagem na utilização deste é necessário o emprego e a integração dos recursos da tecnologia da informação

2.4. Engenharia de *Software*

Engenharia de *Software* envolve várias atividades relacionadas ao desenvolvimento do *software*, desde o surgimento da idéia do desenvolvimento do sistema até a sua desativação.

Uma primeira definição de engenharia de *software* foi proposta por Fritz Bauer [NAU69] et. al., (apud REZENDE, 1999, p. 2), “O estabelecimento e uso de sólidos

princípios de engenharia que possa obter economicamente um *software* que seja confiável e que funcione eficientemente em máquinas reais”.

É considerada como uma solucionadora de problemas inteligente, fazendo valer sua presença ou contratação. Busca gerenciar o desenvolvimento fazendo uso de padrões metodológicos para assim garantir a qualidade e atendimento dos requisitos do cliente.

Conforme PRESSMAN (1995) define Engenharia de *Software* como,

É o estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente um *software* que seja confiável e que funcione eficientemente em máquinas reais, é descendente de engenharia de sistemas e de *hardware*. Abrange um conjunto de três elementos fundamentais (métodos, ferramentas e procedimentos, que possibilita, ao gerente, o controle do processo de desenvolvimento do *software* e oferece ao profissional uma base para a construção de *software* de alta qualidade.

Possui características como a adequação aos requisitos funcionais do negócio do cliente e seus respectivos procedimentos pertinentes, busca atender a efetivação de padrões de qualidade e produtividade em suas atividades e produtos, tem sua fundamentação na tecnologia da informação disponível, viável e oportuna, utiliza planejamento e gestão de atividades, recursos, custos e datas.

Outro conceito de Engenharia de *Software* conforme MARTIN e McCLURE (1991) et. al., (apud REZENDE (1999, p. 04),

É o estudo dos princípios e sua aplicação no desenvolvimento e manutenção de sistemas de *software*, tanto a engenharia de *software* como as técnicas estruturadas são coleções de metodologias de *software* e ferramentas.

Com esses conceitos apresentados identifica-se que Engenharia de *Software* é uma metodologia para desenvolvimento de soluções de *software*, que possibilita utilizar diversas técnicas. Os objetivos da Engenharia de *Software* são o aprimoramento da qualidade dos produtos de *software* e o aumento da produtividade dos engenheiros de *software*, atendendo aos requisitos de eficácia e eficiências, ou seja, efetividade REZENDE (1999).

Na Engenharia de *Software* conforme REZENDE (1999) destaca-se a figura do engenheiro de *software*, conhecido também como criador de soluções, na qual considera-se como tal todo profissional da área de informática ou da ciência da computação que desenvolve soluções profissionais utilizando-se dos recursos de *software*, observando padrões de qualidade requeridos.

A Engenharia de *Software* é uma tecnologia em camadas conforme PRESSMAN (2006), envolvendo ferramentas, métodos, processo e foco na qualidade. Os processos de

softwares formam a base para o controle gerencial de projetos de *software* e estabelecem o contexto no qual os métodos técnicos são aplicados, os produtos de trabalho (modelos, documentos, dados, relatórios, formulários) são produzidos, os marcos são estabelecidos, a qualidade é assegurada e as modificações são adequadamente geridas.

Os métodos de engenharia de *software* fornecem a técnica de como fazer para desenvolver o *software*, abrange um amplo conjunto de tarefas que incluem comunicação, análise de requisitos, modelagem de projeto, construção de programas, testes e manutenção.

As ferramentas de engenharia de *software* fornecem apoio automatizado para os processos e para os métodos. PRESSMAN (2006) comenta que quando as ferramentas são integradas de modo que a informação criada por uma ferramenta possa ser usada por outra, um sistema de apoio ao desenvolvimento de *software*, chamado engenharia de *software* por computador, é estabelecido.

Engenharia de *Software* é o estudo dos princípios e sua aplicação no desenvolvimento e manutenção de sistemas de *software*, ou seja, compreende métodos e controle de gerenciamento, análise, projeto, programação, verificação, testes e manutenção.

2.5. Ciclo de Vida dos Sistemas de Informação

Ciclo de vida de *software* significa uma visão ao longo prazo, que surge quando existe a necessidade de se ter um *software* em uma organização, passando pela manutenção, implantação até a sua desativação. REZENDE (2005, pg. 24) define que um sistema de informação que utiliza recursos da tecnologia da informação pode ter um ciclo de vida curto, de no máximo cinco anos, quando não sofre implementações.

O ciclo de vida do *software* especifica as etapas pelas quais este passa, desde o momento em que é concebido até quando é aposentado. A etapa da análise do ciclo de vida do *software* se concentra na definição do problema a ser solucionado, e identifica a forma de solucioná-lo corretamente sem perder tempo.

O ciclo de vida natural abrange as fases: concepção ou criação, construção ou programação, implantação (disponibilização); implementações (pequenos ajustes ou melhorias); maturidade (utilização plena do sistema); declínio. Manutenção; morte ou descontinuidade. Caso as primeiras fases sejam elaboradas de forma errada, a morte do sistema de informação com certeza irá acelerar, principalmente se o sistema focar a gestão estratégica da organização.

REZENDE (2003, pg. 54) afirma que “não existe *software* ‘pronto e acabado’’, pois ao longo de sua vida exigirá: manutenção legal, correções e melhorias e/ou implementações”. Isso caracteriza que o *software* tem que ser flexível as mudanças que ainda poderão surgir.

Muitas pessoas associam o termo *software* ou sistemas aos programas de computadores. *Software* ou sistema não é só um programa, mas sim toda uma documentação envolvida que exige uma documentação bem detalhada necessária para sua correta utilização e futuras manutenções.

O processo de *software* segundo SOMMERVILLE (2003, pg. 07) “é um conjunto de atividades e resultados associados que geram um produto de *software*”. Quem executa estes processos normalmente é o engenheiro de *software*. Destacam-se quatro atividades fundamentais utilizados em todos os processos de software:

- Especificação do *software*: Onde ocorre a definição da funcionalidade e as restrições em suas operações;
- Desenvolvimento do *software*: O desenvolvimento do software deve ocorrer de acordo com as suas especificações;
- Validação do *Software*: Precisa ser validado para garantir que funcione de acordo com que o cliente exigir;
- Evolução do *Software*: O *software* precisa e deve evoluir sempre para atender as mudanças do cliente.

Para atender esses processos encontram-se vários modelos de processo de *software*. A idéia do modelo é controlar e construir um *software* visando melhorar os padrões dele para assim garantir que se desenvolva um *software* que atenda o máximo das expectativas do cliente.

Entre todos os processos de *software* diferentes, encontramos atividades em comuns entre eles, tais como: especificação de *software*, projeto e implementação de *software*, validação de *software* e evolução de *software*. Sendo assim, cada processo ocorre por etapas, aumentando a possibilidade de se desenvolver um *software* de acordo com as necessidades dos usuários e/ou cliente.

Os modelos mais encontrados para controlar esses processos são: em cascata, espiral e incremental.

- **Modelo em Cascata:** Foi o primeiro modelo criado a partir de outros processos de engenharia por Royce em 1970, possui sua principal característica uma seqüência em cascata de uma fase para outra. Nesse modelo pode-se analisar os requisitos, o que foi projetado, desenvolvido, testado e distribuído no sistema. Conforme BOGGS (2002) o próprio nome sugere, essa cadeia não aceita o fluxo inverso das operações – a água não sobe. O modelo em cascata é apresentado na figura 2.

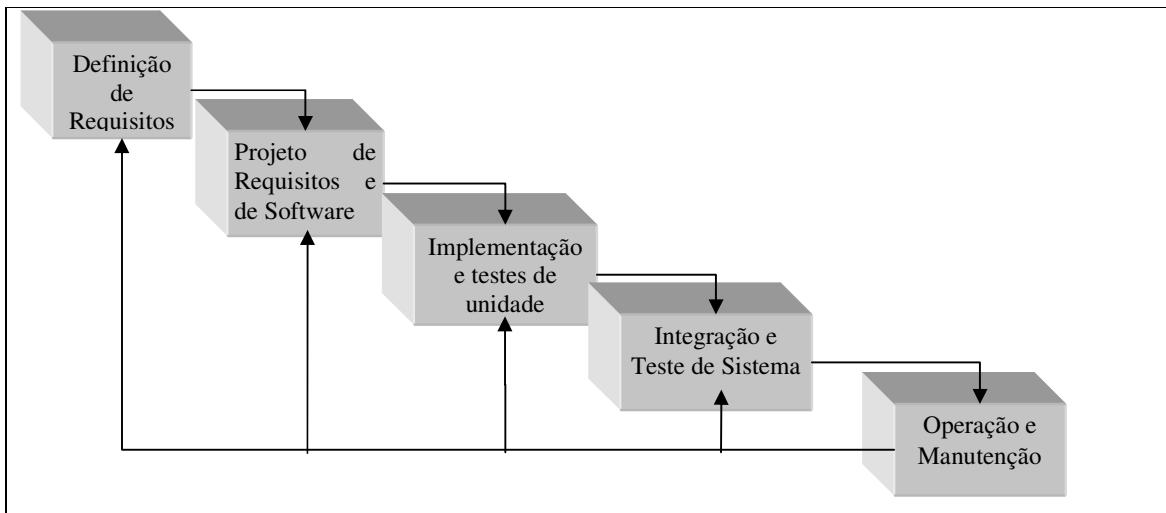


Figura 2: Modelo em cascata.

Fonte: Engenharia de Software, SOMMERVILLE (2003, pg.38).

- **Modelo Incremental:** Foi desenvolvido por Mills em 1980, com o objetivo de reduzir o retrabalho. SOMMERVILLE (2003) destaca que com o modelo incremental é possível fazer com que os clientes identifiquem através de um esboço as funções que o sistema irá oferecer. O cliente identificará quais funções são mais importantes e quais são menos importantes para o cliente. O modelo incremental é apresentado na figura 3.

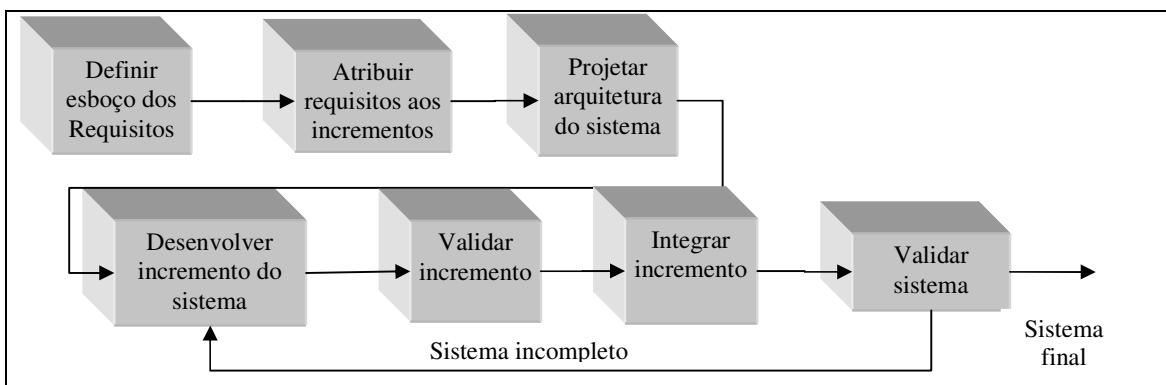


Figura 3: Modelo Espiral.

Fonte: Engenharia de Software, SOMMERVILLE (2003, pg.43).

- Modelo em Espiral: É o modelo mais conhecido, criado por Boehm (1988). Conforme PRESSMAN (2006) este modelo combina a natureza interativa da prototipagem com os aspectos controlados e sistemáticos do modelo em cascata. Possui varias divisões sendo que cada uma tem um objetivo de finalizar um processo do sistema. Uma das considerações mais importantes que difere este modelo dos outros é que se torna explicita a preocupação com o risco. Tornando-o risco menor para que não ocorra nada de errado. Este modelo apresenta de uma forma bem visível os pontos que deverão se levados em consideração para não ocorrer falhas, deixando o cliente mais seguro do *software* que irá adquirir. O modelo em espiral é visto na figura 4.

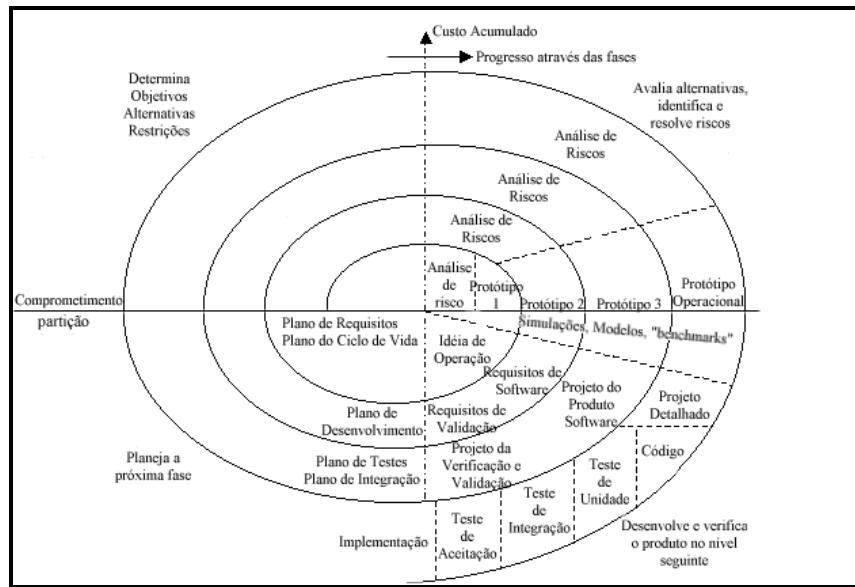


Figura 4: Modelo Espiral.

Fonte: Engenharia de *Software*, SOMMERVILLE (2003, pg. 45).

Após analisado os modelos acima, pode-se identificar primeiramente que é necessário analisar qual é o tipo de cliente que a organização irá atender, qual é o tipo de sistema que se pretende desenvolver, para assim pode definir qual modelo usar. Sendo que cada modelo, por mais que pareça atuar de forma diferente dos outros, o objetivo continua o mesmo, manter o controle e padrões, tanto de desenvolvimento como na documentação.

Os sistemas precisam ser flexíveis a mudanças, para que receba otimizações, reuso e adaptações que futuramente serão solicitadas pelo cliente, fazendo assim com que seu ciclo de vida seja mais prolongado.

2.6. Metodologia de desenvolvimento de sistemas de informação.

O objetivo de uma metodologia no desenvolvimento de sistemas constitui-se em gerenciar e organizar as atividades a serem realizadas durante todo o desenvolvimento de sistemas.

REZENDE (2005, pg. 67) define que,

Uma metodologia completa constitui-se em uma abordagem organizada para atingir um objetivo, por meio de passos preestabelecidos. É um “roteiro” ou um processo dinâmico e interativo para desenvolvimento estruturado de projetos de sistemas de informação (inclusive de *software*) visando à qualidade, produtividade, efetividade e inteligência desses projetos.

Quando é seguida a risca uma metodologia, é possível visualizar através de uma documentação e de uma simbologia as funcionalidades e os serviços que o sistema fornecerá ao cliente.

Conforme SCOTT (1998, pg. 56) “uma metodologia é a abordagem que você utiliza para desenvolver sistemas, enquanto notação é uma coleção de símbolos que você utiliza para documentar trabalhos”. Sendo assim, a metodologia auxilia em cada etapa do desenvolvimento do *software*, controlando e mantendo um padrão de documentação que irá ajudar a minimizar erros.

Para OLIVEIRA (1999, pg. 8) “[...] uma metodologia de sistemas comumente identifica as principais atividades [...] a serem executas e indica quais pessoas [...] devem estar envolvidas em cada atividade e que papel deverão desempenhar”.

Desde que seja seguida a risca, a metodologia permite gerenciar em todas as etapas do desenvolvimento do *software*, garantindo a qualidade, entrega do produto acabado dentro do prazo e dentro dos custos previstos.

De acordo com OLIVEIRA (1999, pg. 20) “uma boa metodologia, aliada a um bom trabalho de análise, projeto e construção, tem tudo para gerar um sistema de informação excelente relação custo-benefício”.

As metodologias conhecidas pelos desenvolvedores são a Orientada a Objetos e a Estruturada. A análise estruturada tem o principal objetivo a aproximação do usuário. Este tipo de análise é baseado em fluxo de execução do sistema, com isso se torna difícil haver mudanças após a implementação. Seu uso não é natural, o que dificulta e muita as mudanças.

DAVIS (1994, pg. 26) ainda comenta que:

Ela é gráfica consistindo em mais figuras do que palavras. Ela é lógica, descrevendo um modelo independente de implementação, que será desenvolvido para o usuário. É uma abordagem passo-a-passo para o desenvolvimento de sistema, começando com o projeto lógico, e gradativamente partindo para o projeto físico.

Já a análise orientada a objeto (AOO) é baseada na decomposição do sistema de acordo com os objetos (entidades de dados). A metodologia abordada neste projeto é a orientada a objeto, por ter uma notação gráfica que consiste na construção de um modelo de um domínio da aplicação e na posterior adição a este dos detalhes de implementação durante o projeto de um sistema. Estas metodologias possuem as seguintes etapas: análise, projeto do sistema, projeto dos objetos e implementação. Estes conceitos podem ser aplicados durante todo o ciclo de vida do desenvolvimento do *software*, desde a análise até o projeto e a implementação.

A Técnica de Modelagem de Objeto faz uso de três tipos de modelos conforme BLAHA (2006) para descrever um sistema: o modelo de classes representa os aspectos estáticos, estruturais de dados de um sistema; o modelo de estados representam os aspectos temporais, comportamentais, de controle de um sistema; modelo de interações representam a colaboração dos objetos individuais, os aspectos de interações de um sistema.

As metodologias e ferramentas disponíveis para desenvolver *softwares* são várias, todas com um único objetivo, auxiliar desenvolvimento com o objetivo de documentar, facilitar o entendimento do sistema.

2.7. Processo Unificado (*Unified Process*)

O Processo Unificado tem o objetivo de conduzir da melhor os processos do desenvolvimento de sistemas. Atualmente os sistemas estão caminhando em direção a sistemas cada vez maiores e mais complexos.

Acredita-se que isso se deve, em partes, ao fato de os computadores estarem cada vez mais potentes, mas também é possível perceber pelo uso constante da *Internet*. Mas se a análise for ainda mais crítica identifica-se ainda um fator relevante, o indicador contínuo que faz com que o sistema cresça cada dia mais, que é a necessidade dos clientes.

Atualmente os clientes querem controlar todo o processo da organização, de tal forma que possam gerenciar toda a informação gerada e assim traçar estratégias para lançar novos produtos, tomar decisões importantes sobre o rumo da organização. Com base na idéia de se desenvolver sistemas de grande porte, mas com qualidade, segurança, com ótima usabilidade,

percebe-se a necessidade de se utilizar metodologias para controlar e gerenciar bem os processos do desenvolvimento do sistema.

SCOTT (2003, pg. 19) define o conceito de Processo Unificado como sendo:

...um conjunto de atividades executadas para transformar um conjunto de requisitos do cliente em um sistema de *software*. Entretanto, o Processo Unificado também é uma estrutura genérica de processo que pode ser customizado adicionando-se ou removendo-se atividades com base nas necessidades específicas e nos recursos disponíveis para um projeto.

O intuito de se utilizar uma metodologia é organizar o projeto, dividindo-o em subprojetos para, onde cada subprojeto representa uma iteração, que deverá ser executada conforme o pré-estabelecido no projeto principal.

No núcleo da UML, está o modelo, o qual no contexto de um processo de desenvolvimento de *software* é uma simplificação da realidade que ajuda a entender alguns aspectos complexos inerentes a sistemas de *software* SCOTT (2003).

Com o uso correto das ferramentas de auxilio para modelar no desenvolvimento do *software* ajuda a controlar os processos, possibilitando diminuir os riscos, onde é necessário respeitar prazos, orçamentos e a adequação com as necessidades do cliente, para assim evitar o fracasso em projetos de *software*.

Para modelar o *software* o Processo Unificado faz uso extensivo da *Unified Modeling Language* (UML).

2.7.1. Princípios Chaves do Processo Unificado

No Processo Unificado se determinou princípios chaves do processo como: Dirigido por casos de uso, Centrado em Arquitetura e Iterativo e Incremental SCOTT (2003).

- Dirigido por casos de uso: Este termo dirigido por caso de uso significa que a toda a interação do trabalho será dada através de diagramas de caso de uso, iniciando desde a captação inicial e negociação de requisitos até a aceitação do código. O caso de uso é um texto narrativo ou gabarito que descreve uma função ou característica do sistema do ponto de vista do usuário, o caso de uso é escrito pelo usuário e serve como base para criação de um modelo de análise mais abrangente. Conforme SCOTT (2003), um caso de uso é uma seqüência de ações, executadas por um ou mais atores (pessoas ou entidades não-humanas

fora do sistema) e pelo próprio sistema, que produz um ou mais resultados de valor para um ou mais atores.

- Dirigido por caso de uso: Os casos de uso são identificados na fase do levantamento de requisitos, onde o engenheiro de *software* identifica os usuários que estarão envolvidos no sistema e quais funcionalidades o sistema precisará efetuar. Com o uso de diagrama de caso de uso, fica mais fácil ilustrar e identificar as atividades de cada usuário, possibilitando mostrar ao cliente quais funcionalidades o sistema estará gerenciando.

- Centrado em Arquitetura: É a organização fundamental do sistema como um todo, desde questões como desempenho, escalabilidade, reuso e restrições econômicas e tecnológicas. Fornece uma base sólida para a construção do *software*, melhora compreensão do sistema e organização do desenvolvimento, a arquitetura representa a forma, enquanto que os casos de uso representam funcionalidades. Os conceitos de centrado em arquitetura de *software* encontrados são vários, dependendo de quem o define. SCOTT (2003, pg. 22), define como uma construção de uma casa ou outra obra mas que envolve toda ela desde os alicerces.

Processo Unificado especifica que a arquitetura do sistema em construção como um alicerce fundamental sobre o qual ele se erguerá deve ser uma das principais preocupações da equipe de projeto e, também, que a arquitetura, em conjunto com os casos de uso, deve orientar e explorar de todos os aspectos do sistema.

- Centrado em Arquitetura: Com este conceito pode-se entender que a mesma preocupação que surge para um engenheiro civil ao construir uma casa surgirá para o engenheiro de *software*, os riscos precisam ser eliminados, ou caso contrário a construção de ambos não representará segurança, levando esses projetos à morte em curto prazo.

A descrição da arquitetura tem como propósito, em primeiro lugar e acima de tudo, facilitar o entendimento da arquitetura do sistema em construção. Com o uso de ferramentas que auxiliam a modelagem através da construção de diagramas é possível ter uma visão geral das necessidades e problemas a serem enfrentados no desenvolvimento do *software*. Uma arquitetura de *software* bem construída oferece “estrutura” sólida na qual os componentes podem residir e trabalhar uns com os outros de modo elegante, ao mesmo tempo em que facilita as equipes que estão trabalhando na construção de outros sistemas identificarem oportunidades para possível reuso de qualquer um dos componentes ou de todos.

- Iterativo e Incremental: Nesta seção identificamos que durante o desenvolvimento do projeto para que ocorra um maior controle de seus processos se torna mais prático dividi-lo em projetos menores.

SCOTT (2003, pg. 24) define que “Uma interação é um mini-projeto que resulta em uma versão de sistema liberada interna ou externamente. Supõe-se que essa versão ofereça uma melhora incremental sobre a anterior, motivo pelo qual interação é chamada de incremento”. A parte interações se refere às etapas do fluxo de trabalho, e incremento, ao avanço no desenvolvimento do produto. Para que se tornem mais efetivas, as interações devem ser controladas, nesta situação devem ser executadas de modo planejado, neste momento entende-se porque deve ser considerados mini-projetos.

Através de cada interação os desenvolvedores procuram identificar e especificar os casos de uso relevantes, desenvolvem o projeto de acordo com a arquitetura escolhida, implementam o projeto em componentes e verificam se os componentes satisfazem os casos de uso.

Desenvolver a arquitetura de um modo interativo e incremental possibilita ainda no inicio do projeto fazer alterações que são necessárias, fazendo com não se torne um custo considerável no final do projeto.

Uma das vantagens encontradas é antecipar e corrigir os problemas é a redução do retrabalho que ocasiona em manutenções, sem contar nos custos que podem gerar, e que em muitas vezes pode se tornar viável a continuação do projeto podendo vir a não atender os requisitos do cliente.

2.7.2. As Fases do Processo Unificado

O ciclo de vida do *software* está dividido em fases, ao final de cada fase é disponibilizada uma versão que é liberada ao cliente. Segundo SCOTT (2003, pg. 26) “Uma fase é simplesmente o tempo decorrido entre dois marcos principais, em que gerentes tomam decisões importantes sobre se prosseguem com o desenvolvimento e, se este for o caso, o que é necessário em relação ao escopo, orçamento e cronograma do projeto”. Cada fase do ciclo de vida existe quatro fases do processo unificado.

As fases são concepção, elaboração, construção e transição, vistas na figura 5 e na qual estão definidas a seguir:

- **Concepção:** O principal objetivo dessa fase é estabelecer a viabilidade do sistema proposto. Durante esta fase são definidas as seguintes tarefas: o escopo do sistema, esboço da arquitetura. Além de ajudar a identificar riscos críticos e determinar quando e como o projeto os abordará, auxilia a iniciar uma análise econômica do projeto, mostrando se o projeto vale à pena, com base em estimativas iniciais de custo, esforço, cronograma e qualidade do produto (SCOTT, 2003).

- **Concepção:** Nesta fase os principais interessados no projeto, precisam saber e concordar com o escopo do sistema proposto. É preciso saber o que o sistema irá fazer e se o que ele irá fazer de fato atenderá os requisitos do cliente.

SCOTT (2003, pg. 27), define que “O principal marco associado com a fase de concepção é chamado Objetivos do Ciclo de Vida. As indicações de que o projeto o atingiu incluem:

- A arquitetura candidata equaciona claramente um conjunto de requisitos críticos de alto nível;
- “A análise econômica do projeto é sólida o suficiente para justificar a continuação do desenvolvimento”.
- **Elaboração:** Um dos objetivos principais desta fase é delimitar o escopo do projeto, definindo como o sistema será utilizado por cada usuário, através dos casos de uso mais relevantes. A partir do escopo é possível identificar os custos, prazos e até retorno do investimento. Outra tarefa executadas durante a elaboração é a análise dos riscos, que precisam ser trabalhados e solucionados para que não se torne um problema futuramente, que possa gerar custos.
- **Construção:** Esta fase é caracterizada pela capacidade de construir o sistema. Durante a construção a equipe de projeto busca executar tarefas no sistema de modo iterativo e incremental, para assim analisar a viabilidade, usabilidade do sistema. Segundo SCOTT (2003, pg. 22), “O principal marco associado à fase de construção é chamado Capacitação Operacional Inicial. O projeto terá atingido esse marco se um conjunto de clientes beta tiver um sistema operacional mais ou menos completo nas mãos”.

Esta fase terá como resultado uma versão que será disponibilizado ao cliente para testes em seu ambiente, podendo analisar o sistema e suas funcionalidades.

- **Transição:** É a ultima etapa, onde é feita a entrega da versão oficial do *software* ao cliente. Segundo SCOTT (2003, pg. 29) “O principal macro associado à fase de transição é

chamado “Liberação do Produto”. Claro que esta fase também se concentra em corrigir defeitos e modificar o sistema para eventuais correções dos problemas a serem identificados.

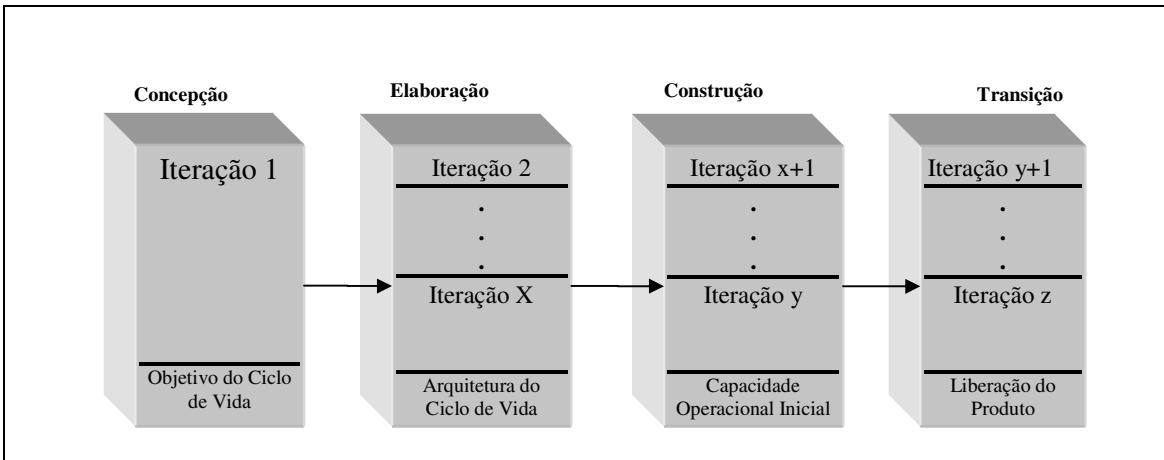


Figura 5: Fases e Marco Principais.

Fonte: O Processo Unificado Explicado, SCOTT (2003, pg.27).

2.8. Rational Unified Process (RUP)

O RUP (*Rational Unified Process*) é um processo de *software* da Engenharia de *Software*. O RUP pertence a *Rational Software*. É classificado como uma instância específica e altamente detalhada do processo unificado. MATOS (200) comenta que “o Processo Unificado proposto pela Rational (*Rational Unified Process – RUP*) foi criado para apoiar o desenvolvimento orientado a objetos, fornecendo uma forma sistemática para se obter reais vantagens no uso da Linguagem de Modelagem Unificada (*Unified Modeling Language – UML*)”. O RUP está fundamentado em três princípios básicos: orientação a casos de uso, arquitetura e iteração.

Segundo SCOTT (2003, p. 19), “é uma versão especializada do Processo Unificado que adiciona elementos a estrutura genérica”. O RUP é organizado em fases e interações, as fases representam o início e a nítida conclusão de uma atividade da Engenharia de *Software*, se atentando a quatro fases como Concepção, Elaboração, Construção e Transição.

O RUP constitui de um conjunto de páginas HTML, tutoriais da ferramenta, manuais para o usuário poder desenvolver utilizando as técnicas dentro do padrão do processo unificado. O RUP é uma junção de várias técnicas, enfatizando duas características encontradas em outros processos de *software* como a gerência de riscos e a modelagem da arquitetura do *software*, por ser uma ferramenta que disponibiliza vários recursos o RUP é um

processo aplicável a grandes projetos, mas podendo ser aplicado em projetos menores, devido a sua customização possibilitar adaptações para utiliza-lo em vários outros projetos.

2.9. *Unified Modeling Language (UML)*

Para que um *software* atenda os requisitos do cliente e atenda as expectativas da organização a qual será será desenvolvido, é preciso utilizar uma metodologia no desenvolvimento desde seu ínicio. Percebemos que isto ocorre já na primeira entrevista, então é preciso utilizar toda uma documentação, para vizualizar em diversas situações, documentar para oder controlar o desenvolvimento para assim extrair do cliente o realmente quer que o sistema faça, para que não ocorra um desconforto durante as outras etapas do desenvolvimento ou até se desenvolver um produto que não atenda a necessidade do cliente.

Para que isto ocorra conforme GUEDES (2004) é utilizados métodos para organizar os processos para que possa ser apresentado através de uma simbologia que facilitará o entendimento do sistemas. Um *software* desenvolvido sem nenhuma metodologia é a mesma coisa que caminhar sem saber onde quer chegar, o *software* fica a merce do programador, sendo que só ele entende, só ele da a manutenção, reduzindo assim o tempo de vida do sistemas, tornando seu custo alto, e não atendendo a necessidade do cliente.

A metodologia deve ser empregada em todos os projetos com a idéia de padronizar o desenvolvimento do *software* facilitar sua manutenção, e desenvolvê-lo de acordo com os requisitos do cliente Para isto é utilizado a modelagem com o intuito de compreender melhor o sistema que esta sendo desenvolvido. No sentido mais amplo, um modelo é a simplificação da realidade.

Em meados da década de 90 se utilizavam com mais intensidade três metodologias para o desenvolvimento do *software*, sendo que cada uma delas eram similares, que auxiliavam em todas as estapas do desenvolvimento do *software*, tendo o método de Booch, o método OMT (*Object Modeling Technique de Jacobson*) e o método OOSE (*Object-Oriented Software Engineering*) de Rumbaugh. Durante a decada de 90 foram estas as três metodologias de modelagem orientda a objetos mais usadas entre os profissionais da área de desenvolvimento de *software*.

Como cada uma dessas metodologias tinham um objetivo em comum que é o de modelar o *software*, percebeu-se que ambas estavam se deslocando na mesma direção com as mesmas caracteristicas, após uma iniciativa conjunta de Grady Booch, Dr. James Rumbaugh,

Ivar Jacobson, Rebecca Wirfs-Brock, Peter Yourdon e muitos outros, sendo então que responsáveis por cada uma delas e através do apoio da *Rational Software*, que financiou e incentivou o projeto houve a união das metodologias dando origem a UML.

Em 1996 a união dos três amigos como popularmente ficou conhecido, Rumbaugh, Booch e Jacobson surgi a primeira versão da UML – *Unified Modeling Language* GUEDES (2004). Após sua primeira versão várias empresas atuantes na área de modelagem e desenvolvimento de *software* passaram a contribuir fornecendo sugestões para melhorar e ampliar cada vez mais a linguagem. Em 1997 a UML foi adotada pela OMG (Object Management Group ou Grupo de Gerenciamento de Objetos) ganhando mais força e se tornando uma linguagem padrão de modelagem. Hoje estamos na versão 2.0 que segundo GUEDES (2004) esta nova versão traz grandes novidades em relação à estrutura geral da linguagem principalmente com relação a abordagem de quatro camadas e a possibilidade de se desenvolver “perfis” particulares a partir da UML. Segundo estudos UML não é um método é uma linguagem de modelagem designada para especificar, visualizar, construir e documentar um sistema.

Um conceito definido por GUEDES (2004, pg. 17) é que:

A UML não é uma linguagem de programação e sim uma linguagem de modelagem, cujo objetivo é auxiliar os engenheiros de *software* a definir as características do *software*, tais como requisitos, seu comportamento, sua estrutura lógica, a dinâmica de seus processos e até mesmo suas necessidades físicas em relação ao equipamento sobre o qual o sistema deverá ser implantado.

A linguagem de modelagem é a notação que o método utiliza para expressar projetos enquanto que o processo indica quais passos seguir para desenvolver um projeto.

Conforme PILONE (2006, pg. 5) “a modelagem é um meio de documentar idéia, relacionamentos, decisões e requisitos numa notação bem definida que pode ser aplicada a muitos domínios diferentes”. O modelo UML é feito de um ou vários diagramas, utilizados para representar graficamente situações e as relações entre essas situações. Os diagramas são representações gráficas de um conjunto de elementos, que permitem visualizar o sistema sob diferentes perspectivas. A UML divide os diagramas em duas categorias: diagramas estruturais e diagramas comportamentais. Os diagramas estruturais normalmente são utilizados para documentar, visualizar, especificar os aspectos estáticos de um sistema.

Outro conceito sobre UML é do MELO (2005, pg. 33):

A UML proporciona uma forma padrão para a preparação de planos de arquitetura de projetos de sistemas, incluindo aspectos conceituais tais como processos de negócios e funções do sistema, além de itens concretos como as classes escritas em determinada linguagem de programação, esquemas de bancos de dados e componentes de *software* reutilizáveis.

Os aspectos estáticos de um sistema de *software* abrangem a existência e a colocação de itens como classes, *interfaces*, colaborações, componentes e nós, enquanto que os diagramas comportamentais são usados para visualizar, especificar, construir e documentar os aspectos dinâmicos de um sistema que é a representação das partes que sofrem alterações, como por exemplo o fluxo de mensagens ao longo do tempo e a movimentação física de componentes em uma rede.

A UML suporta o desenvolvimento iterativo e incremental. Desenvolvimento iterativo e incremental é o processo de desenvolvimento de sistemas em pequenos passos. Uma iteração é um laço de desenvolvimento que resulta na liberação de um subconjunto de produtos que evolui até o produto final percorrendo as seguintes atividades: Análise de requisitos, Análise, Projeto, Implementação, Testes.

2.9.1. Diagrama de Casos de Uso

Este diagrama é inserido assim que é feito o levantamento de requisitos e é utilizado como material de consulta durante todo o processo de modelagem. Esta representação gráfica tem o objetivo de uma forma simples apresentar uma idéia geral de como o sistema irá funcionar.

PILONE (2006, pg. 77) define “que os casos de uso são uma maneira de capturar a funcionalidade do sistema e todos os requisitos de UML”. Nesta etapa se faz necessário extrair do usuário, os tipos de usuários que irão interagir com o sistema, quais as funções destes usuários com o sistema assim como outras informações precisas para elaboração do diagrama, aumentando o entendimento do que o sistema irá precisar fazer. O engenheiro de *software* deve analisar o documento de requisitos ou conjunto de casos de uso e projetar o sistema antes que os programadores o implementem em uma linguagem de programação em particular.

Conforme DEITEL (2007, pg. 47) o diagrama de caso de uso “modelam as interações entre um sistema e suas entidades externas (atores) em termos de caso de uso (capacidade do sistema, como ‘Visualizar saldo em conta’, ‘Sacar dinheiro’ e ‘Depositar fundos’)”.

Este diagrama é apresentado ao usuário juntamente com um protótipo que irá auxiliar no entendimento do funcionamento do sistema. Durante esta etapa ocorrem problemas principalmente com a interpretação do que o usuário espera do sistema, o engenheiro de *software* precisa e muito ficar atento a todos os detalhes, apresentar modelos que possam contribuir para o entendimento da construção do *software*.

Este diagrama de caso de uso possui dos itens principais: atores e caso de uso assim como mostra a figura 6:

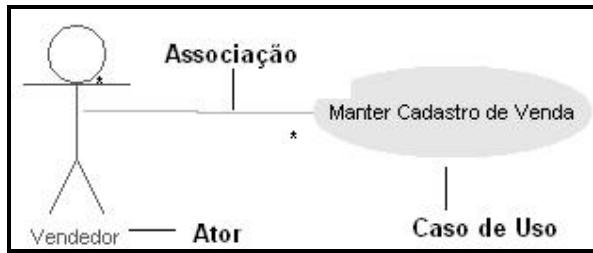


Figura 6: Modelo de diagrama de caso de uso
Fonte: Guilherme Post adaptado de GUEDES (2004).

Os atores podem ser algo ou alguém, desde que tem participação no sistema. O ator é a representação de vários usuários que poderão utilizar o sistema. Mas o ator também pode ser outro *software* que venha a interagir com o sistema. Os casos de usos referem-se aos serviços, tarefas ou funções que podem ser utilizadas de alguma forma pelo usuário do sistema, como por exemplo: emissão de relatórios, cadastro de vendas ou de produtos. São utilizados para expressar e documentar os comportamentos pretendidos para as funções do sistema.

2.9.2. Diagrama de classe

O diagrama de classes é o mais importante e o mais utilizado diagrama da UML. Permite a visualização das classes que compõem o sistema e seus respectivos atributos e métodos. É através deste diagrama que é feita a modelagem para sistemas orientado a objeto, apresentando assim qual classe pertencerá ao sistema. Para DEITEL (2007, pg. 47) os diagramas de classe “modelam as classes ou os ‘blocos de construção’ utilizados em um sistema, esse diagrama nos ajudam a especificar os relacionamentos estruturais entre partes do sistema”.

O diagrama de classes é composto por classe, atributo e método, assim como apresenta a figura 7:

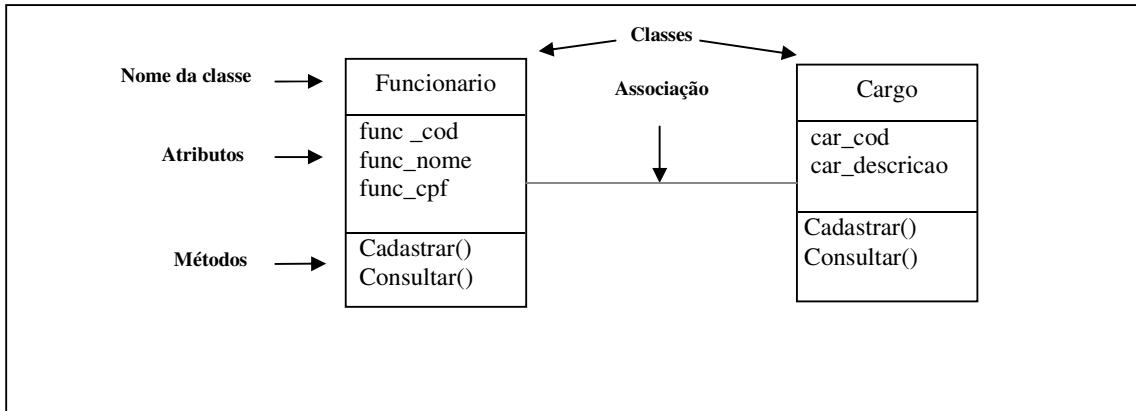


Figura 7: Modelo de diagrama de classe

Fonte: Guilherme Post adaptado de GUEDES (2004).

2.9.3. Diagrama de Seqüência

O diagrama de seqüência baseia-se no diagrama de caso de uso. Este diagrama determina a seqüência em que irão ocorrer os eventos, a forma qual estarão interagindo e quais métodos serão disparados entre os objetos. Sendo que apresenta vários diagramas de seqüência mesmo havendo apenas um diagrama de caso de uso. Conforme SCOTT (1998, pg.65) define que usando o diagrama de seqüência um dos mais importantes aspectos é

Primeiro por que possibilita testar o seu projeto, por ser uma abordagem mais formal que o teste de cenário de utilização. Segundo, é uma excelente forma de documentar seu projeto. Terceiro, são uma excelente forma de detectar gargalos no seu projeto.

Esse diagrama é composto atores, objetos, linha vertical tracejada, conhecida também como linha da vida, foco de controle ou ativação que indica o período em que um determinado objeto está participando ativamente do processo, mensagens ou estímulos são utilizados para demonstrar ocorrências de eventos e mensagens de retorno que identifica a resposta a uma mensagem para o objeto ou ator que a chamou, condições ou condições de guarda indicam que uma mensagem só poderá ser enviada a um objeto se uma determinada condição for verdadeira.

Ao lado dos atores na parte superior do diagrama possuem caixas ou caixas de invocação de métodos que representa o objeto e as linhas horizontais que são as interações onde são enviadas as mensagens entre as linhas de vida dos objetos.

A figura 8 é um exemplo de interação entre objetos representada por um diagrama de seqüência apresentada na pagina 43.

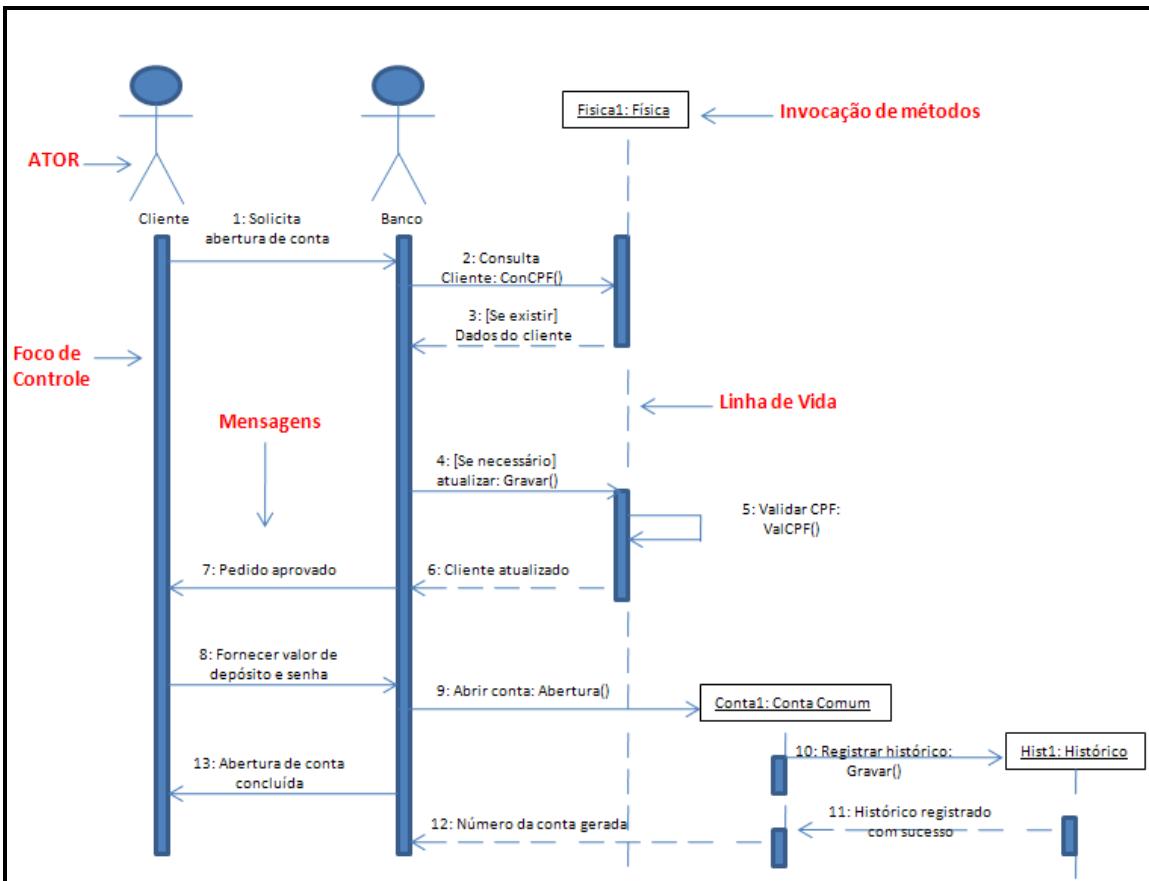


Figura 8: Modelo de diagrama de seqüência

Fonte: GUEDES (2004).

2.9.4. Diagrama de Colaboração

O diagrama de colaboração é utilizado para representar a interação do ator com os casos de uso, as classes, interfaces e outros elementos que compõem o sistema. As colaborações são empregadas para especificar a realização de caso de uso e operações e para fazer a modelagem de mecanismos significativos da arquitetura do seu sistema.

Um conceito de diagrama de colaboração definido por BOOCH (2005) é “uma sociedade de classes, interfaces e outros elementos que trabalham em conjunto para fornecer algum comportamento cooperativo maior do que a soma de todas as suas partes”. A colaboração é como um classificador incluindo classe, interface, componente, nó ou caso de uso, ou uma operação. Toda colaboração deve ter um nome que possa diferenciar uma das outras.

Um diagrama de colaboração é composto por ator que fará interação com a interface (*forms*), que estará relacionada com classe de controle (*controls*) na qual fará conexão com as

classes *entitys*. O diagrama de colaboração é apresentado na figura 9, apresentando cadastro de fornecedores.

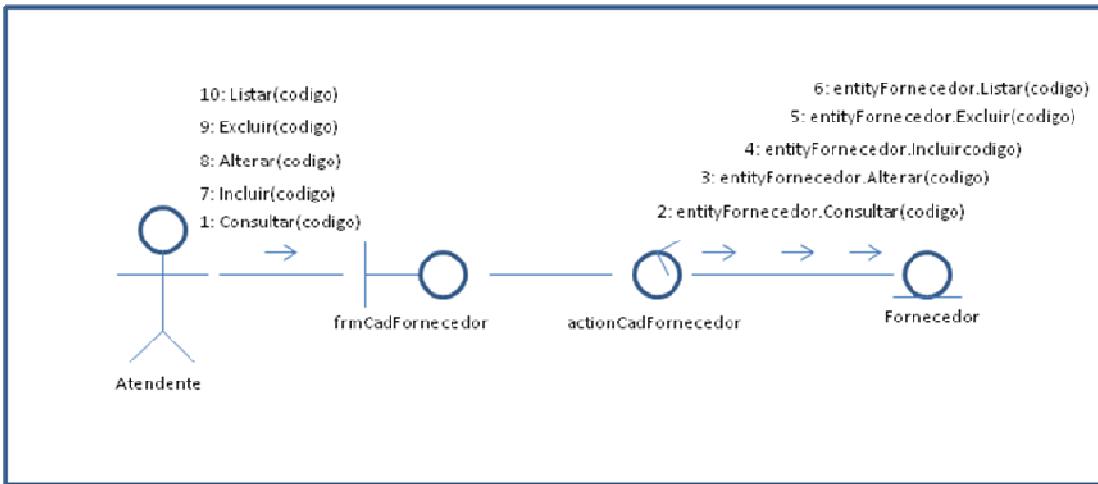


Figura 9: Modelo de Diagrama de Colaboração

Fonte: Adaptado de BOOCH (2005).

As colaborações não apenas nomeiam os mecanismos do sistema, elas também servem como a realização de casos de uso e operação.

2.10. Levantamento de Requisitos

Esta etapa é considerada como uma das mais importantes no desenvolvimento de *software*. É neste momento que é preciso se extrair o maior numero de informações possíveis e de uma forma clara e precisa, para assim poder desenvolver um sistema que atenda os requisitos do cliente.

REZENDE (2005, pg. 65) afirma que esta fase precisa ser bem definida e “sua importância esta relacionada com o sucesso do projeto de sistemas de informação. Durante o levantamento de requisitos é preciso que haja uma boa troca de informações entre usuário e engenheiro de *software* para assim buscar compreender as necessidades do usuário e o que ele deseja que o sistema a ser desenvolvido realize. O levantamento de requisitos inicia-se através de entrevistas, onde o engenheiro de *software* busca entender como funciona atualmente o processo a ser informatizado e quais serviços o cliente precisa que o *software* forneça.

Segundo GUEDES (2004, pg. 19):

Devem ser realizadas tantas entrevistas quantas forem necessárias para que as necessidades do usuário sejam compreendidas. Um dos principais problemas enfrentados na fase de Levantamento de Requisitos é o de comunicação. A comunicação constitui-se em um dos maiores desafios da Engenharia de *Software*, caracterizando pela dificuldade em conseguir compreender um conjunto de conceitos vagos, abstratos e difusos que representam as necessidades e desejos dos clientes e transformá-los em conceitos concretos e inteligíveis.

Para se desenvolver um bom projeto é preciso buscar extrair a maior quantidade de informação durante as entrevistas passadas do usuário ao engenheiro de *software*. Os processos realizados atualmente pela organização a qual será desenvolvido o *software* precisa ser bem compreendido, é necessário realizar entrevistas com todos os envolvidos, *Stakeholder* (Alguém ou algo com interesse ou envolvimento no sistema).

Durante esta fase é importantíssimo a documentação, os dados obtidos devem ser sistematicamente documentados, não podem ser confiados puramente à memória humana. Tendo em vista esta preocupação, foi escolhida para modelagem deste projeto a ferramenta UML e os diagramas de Caso de Uso e de Seqüência, que serão definidos no próximo capítulo.

2.10.1. Modelo de Domínio

Um modelo de domínio descreve a interação entre os objetos. Para facilitar o entendimento do modelo de domínio é preciso identificar e nomear cada objeto, para assim desenvolver um glossário que será possível todos os envolvidos no projeto a entender melhor o sistema.

Um modelo de domínio é desenvolvido para capturar os objetos mais importantes do sistema, esses objetos representam coisas que existem ou eventos que ocorrem no ambiente no qual o sistema trabalha.

O modelo de domínio é descrito através de diagramas UML, normalmente o digrama de classe. Com este digrama é possível apresentar aos *Stakeholders* as classes de domínio e como funciona o relacionamento através de associações.

O BLAHA (2006) apresenta um exemplo do modelo de domínio de uma forma simples, mas de fácil entendimento na figura 9. Este modelo consiste em demonstrar através das classes o sistema de contas de cartão de crédito. Uma instituição pode emitir muitas contas de cartão de crédito, cada uma identificada por um numero de conta. Cada conta possui um limite máximo de crédito, um saldo atual e um endereço de correspondência. A conta atende a um ou mais clientes, que residem no endereço de correspondência. A instituição

periodicamente emite um extrato para cada conta. O extrato lista data de vencimento, encargos financeiros e pagamento mínimo.

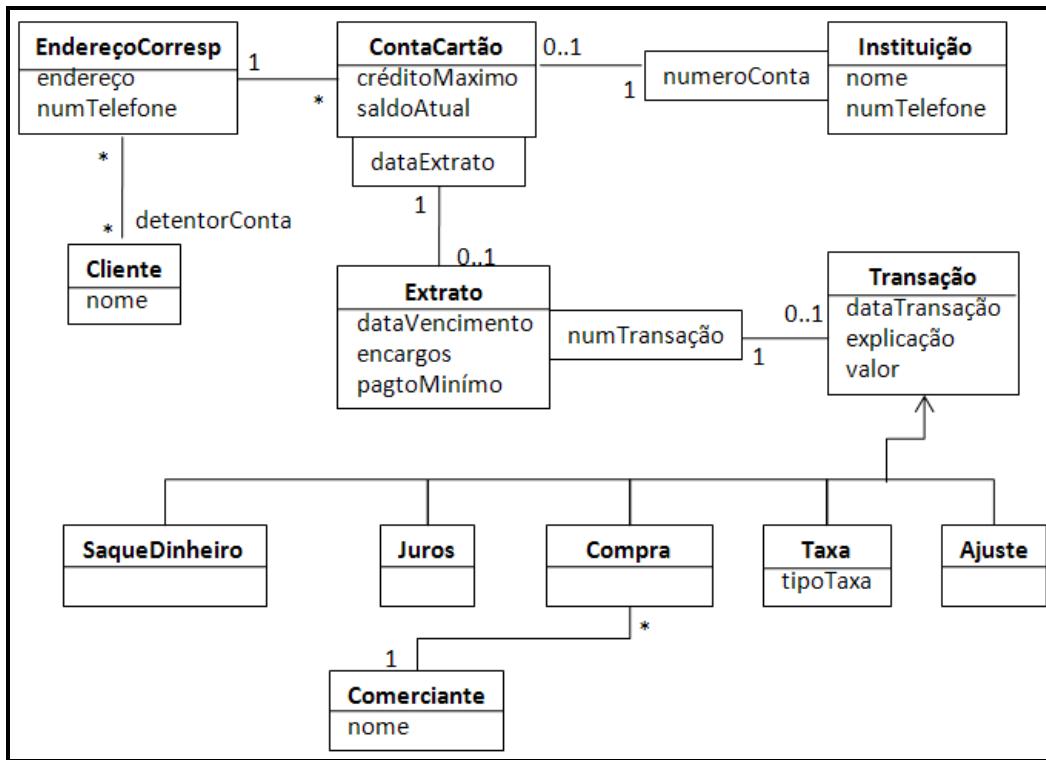


Figura 10: Modelo de domínio

Fonte: BLAHA (2006).

2.10.2. Encontrar Atores e Caso de Uso

Esta atividade tem por objetivo identificar os atores e casos de uso, possibilitando assim criar um modelo de caso de uso.

Desta forma é possível delimitar o sistema a partir do ambiente, é descrito quem irá (ator) interagir com o sistema e quais funcionalidades (caso de uso) são esperadas do sistema.

Os atores são caracterizados por dois tipos segundo GUEDES (2004):

- Atores primários: que usarão o sistema diretamente;
- Atores secundários: que supervisionam e mantêm o sistema, existem apenas para que os atores primários possam utilizar o sistema.

O caso de uso refere-se ao serviço, tarefa ou funções que podem ser utilizadas de alguma maneira pelos usuários do sistema, como emitir relatório ou cadastrar a venda de

algum produto. Os casos de uso são utilizados para expressar e documentar os comportamentos pretendidos para as funções do sistema.

Um caso de uso pode apresentar o um fluxo principal e alternativo. O fluxo principal é o mais importante e representativo curso de eventos. O fluxo alternativo são as variantes do curso básico de erros. A interação do ator com o caso de uso é visto na figura 10.

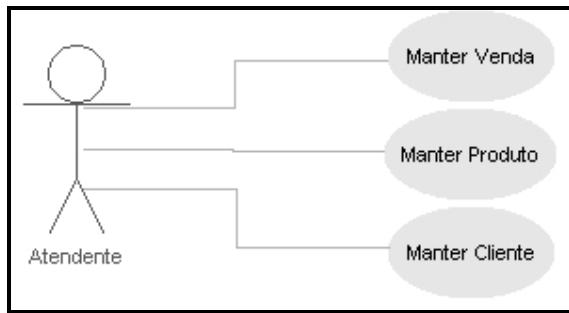


Figura 11: Diagrama de caso de uso
Fonte: Guilherme Post adaptado de GUEDES (2004).

2.10.3. Priorizar Caso de Uso

Nesta etapa são destacados quais casos de uso é preciso da prioridade durante o desenvolvimento do sistema.

Conforme MULLER (2002, pg. 80),

“...deve-se priorizar os objetivos operacionais por sua contribuição à missão básica do sistema. Se o sucesso do sistema exige o alcance bem sucedido do objetivo, este é crítico. Se você poder pular o objetivo com apenas pequenos problemas, ele será marginal. Se o objetivo não possui relação com a missão, ele é um requisito ruim”.

As regras também precisam ser priorizadas e de preferência por sua natureza. Com o modelo de caso de usos, são apresentadas todas as funcionalidades que o sistema precisará fazer, com isso é possível destacar as mais importantes, sendo que precisarão ser implementadas em primeiro lugar, para o correto desempenho do desenvolvimento do *software*.

2.10.4. Detalhamento do Caso de Uso

Nesta etapa é descrita o fluxo de ventos, apresentando como o caso de uso inicia, termina e interage com atores.

Para GUEDES (2004, pg. 47) “Os casos de uso costumam ser documentados, fornecendo instruções em linhas gerais como será seu funcionamento, quais atividades deverão ser executadas, qual evento forçara sua execução, quais Atores os poderão utilizar, quais suas possíveis restrições, entre outras”.

Esta documentação é realizada através de uma linguagem bastante simples, detalhando funções dos casos de usos, quais atores interagem com o mesmo, quais etapas devem ser executadas pelo ator e pelo sistema. Na tabela 2 é apresentado um modelo simples para documentação do caso de uso.

Tabela 2: Documentação de caso de uso abertura de conta.

Nome do Caso de Uso	Abertura de Cota
Caso de Uso Geral	
Autor Principal	Cliente
Atores Secundários	Funcionário
Resumo	Este caso de uso descreve as etapas percorridas por um cliente para abrir uma conta corrente
Pré-Condição	O pedido da abertura precisa ser aprovado
Pós-Condição	É necessário realizar um depósito inicial
Ações do Ator	Ações do Sistema
1.Solicita Abertura de Conta	
	2.Consultar cliente por seu CPF
	3.Se for necessário, gravar ou atualizar o cadastro do cliente. Se o cliente não possuir outras contas deve ser registrado como inativo.
	4.Avaliar o pedido do cliente
	5.Aprovar pedido
6.Escolher a senha da conta	
	7.Abrir conta
	8.Definir cliente como ativo
9.Fornecer valor a ser depositado	
	10.Registrar depósito
	11.Emitir cartão da conta
Restrições/Validações	1.Para abrir uma conta corrente é preciso ser maior de idade 2.O valor mínimo de depósito é R\$5,00

Fonte: GUEDES (2004, pg. 48).

2.10.5. Protótipo de *Interface* do Usuário

É criado um esboço de uma parte do sistema. Tem a finalidade de apresentar através de interfaces o sistema ou parte dele ao cliente, para poder mostrar a aparência e possíveis funcionalidades do sistema.

Está técnica é de fácil aplicação, de rápido desenvolvimento, e de ótimo entendimento do que se é realmente necessário desenvolver. Segundo GUEDES (2004, pg. 20):

...um protótipo normalmente apresenta pouco mais do que uma interface do *software* a ser desenvolvido, ilustrando como as informações seriam inseridas e recuperadas no sistema e apresentando alguns exemplos como dados fictícios de quais seriam os resultados apresentados pelo *software*, principalmente em forma de relatório.

Com a prototipação das *interfaces*, é possível eliminar riscos com o desenvolvimento de um *software* que futuramente pode não ser aceito pelo cliente, por não atender os seus requisitos básicos, descobrindo isso no inicio do desenvolvimento, sem ter que esperar implantar para ver a reação do cliente. Saber o que construir ajuda as organizações que desenvolvem *software* diminuir custo e evitar perda de tempo.

2.10.6. Estrutura do Modelo de Caso de Uso

Os diagramas de caso de uso possuem associações que contemplam nomes especiais como: especialização/generalização, inclusão, extensões.

- Associação: é o relacionamento entre atores ou caso de uso que fazem parte do diagrama, associação é representada através de uma reta que liga o ator ao caso de uso;
- Especialização/Generalização: é o relacionamento entre atores ou caso de uso quem possuem características semelhantes entre si, é representado por uma reta com uma seta mais grossa que indica o caso de uso geral para o qual a seta aponta;
- Inclusão: Utilizada quando existe um serviço, situação ou rotina comum a mais de um caso de uso, é representado por uma reta tracejada contendo uma seta em sua extremidade, ainda contém alguns estereótipos com o texto “*include*”;
- Extensão: São utilizados para criar o cenário, mas que serão utilizados em situações específica, assim que uma determinada condição for satisfeita, é representado também

por uma reta tracejada com uma seta, mas aponta para o caso de uso que utiliza o caso de uso estendido e possui um estereótipo contendo o texto “*extends*”.

2.11. Análise dos requisitos

Após o Levantamento de Requisitos, é feita a Análise de Requisitos, onde as necessidades apresentadas pelo cliente são analisadas. Onde o engenheiro de *software* (ou analista) examina passo a passo os requisitos enunciados pelos usuários, para identificar se as informações coletadas através das entrevistas foram realmente compreendidas. A partir desta etapa são determinadas as reais necessidades do sistema de informação.

Um conceito sobre análise de requisitos segundo GUEDES (2004, pg. 20) é que:

Consiste em determinar se as necessidades dos usuários foram entendidas corretamente, verificando se algum tópico deixou de ser abordado, determinando se algum item foi especificado incorretamente ou se algum conceito precisa ser melhor explicado.

Para realizar esta etapa os engenheiros de *software* precisam reestruturar o modo como as informações são geridas e utilizadas pela empresa e apresentar maneiras de combiná-las e apresentá-las de modo que possam ser mais bem aproveitadas pelos usuários.

Durante análise não existe uma metodologia pré-definida, cada empresa usa a forma que considera melhor para que as funcionalidades e a forma que serão implementadas sejam passadas para o responsável pela codificação.

No projeto de Sistema para uma Agropecuária a análise será desenvolvida, na fundamentação prática, uma engenharia que definirá cada etapa do desenvolvimento.

2.11.1. Analisar Arquitetura

Durante esta etapa a atenção fica voltada para analisar as classes desenvolvidas nas etapas anteriores, podendo assim identificar subsistemas. Com o desenvolvimento poderá surgir a necessidade de se criar novas classes.

Conforme MULLER (2002, pg. 15) “uma arquitetura de sistema é uma estrutura abstrata dos objetos e relacionamentos que compõem um sistema”.

Esta arquitetura é apresentada através de um gráfico onde é visto os atores e casos de uso em subsistemas (subdivisões) para criar o ambiente do sistema.

A idéia de subdividir o sistema em subsistemas nasce do conceito de dividir para conquistar, onde hoje pode ser mais trabalhosos o desenvolvedor criar os subsistemas, mas futuramente quando ocorrer às otimizações ou manutenções, será mais fácil trabalhar com o *software*, sem ter que alterar todo um sistema, e sim tratar o problema localizado.

2.11.2. Analisar Caso de uso

Após definir os casos de uso que compões os subsistemas, é desenvolvido um modelo de análise, com classes de negócios representando coisas, lugares e pessoas, e não componente de *software*. Para desenvolver este modelo são utilizados diagramas de seqüência ou de colaboração, onde ocorre somente à troca de mensagens entre classe e entidade, classe de controle e classe de interface. As classes de análise são identificadas a partir deste modelo descrito na figuras 11,12 e 13:

Classe boundary (classe de interface), são as janelas do sistema.

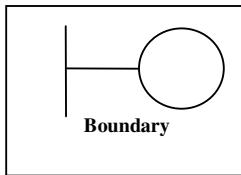


Figura 12: Classe de análise *boundary*.
Fonte: GUEDES (2004).

Classe control (classe de controle), contém as especificações e códigos

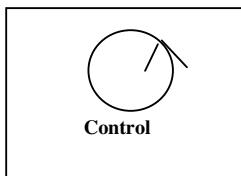


Figura 13: Classe de análise *control*.
Fonte: GUEDES (2004).

Classe entity (classe de entidade), classes de objetos persistentes, são objetos armazenados pelo sistema.

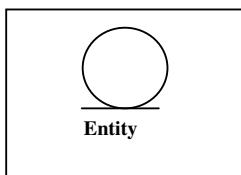


Figura 14: Classe de análise *entity*.
Fonte: GUEDES (2004).

2.12. Projeto

As fases de elaboração enfocam o estabelecimento das bases da arquitetura do projeto, onde é definida a forma e ferramenta que será utilizada para alcançar os objetivos pré-estabelecidos nas etapas anteriores para o desenvolvimento do *software*.

Um dos problemas encontrados nesta etapa segundo SCOTT (1998, pg. 52) é que “grande parte dos desenvolvedores são programadores por natureza, e programadores tipicamente não querem perder tempo desenhando diagramas descrevendo o que estão fazendo. Muitos sistemas não obtiveram sucesso por que não tiveram análise e projetos adequados”.

Para tanto, é impossível construir uma casa sem um projeto, mesmo aquele desenho em um rascunho de papel se torna início de uma documentação, claro sem saber com exatidão aonde quer chegar. Mas é preciso se elaborar um projeto por inúmeros motivos. Alguns motivos: obtenção do custo para desenvolver o sistema, viabilidade do desenvolvimento do *software*, quanto tempo levará para desenvolvê-lo, quais ferramentas necessárias para desenvolver, quais serão os membros da equipe de desenvolvimento que estarão envolvidos, o que o sistema irá fazer e de forma irá atender a necessidade do usuário.

REZENDE (2005, pg. 83) afirma que: “essa fase permite que a equipe envolvida de elabore a solução detalhada do novo projeto do sistema de informação em desenvolvimento, com a real necessidade das informações para a organização.” É nesta etapa que é trabalhada todas as informações obtidas no levantamento de requisitos e na análise, onde serão gerados documentos que definiram os procedimento e métodos a serem empregados para o desenvolvimento do *software*, buscando deixar toda a documentação de uma forma bem clara e de fácil percepção do que é realmente necessário se desenvolver.

No projeto de *software* o processo de entrada fica sendo como os documentos de especificação de requisitos e gera como saída um conjunto de modelos e artefatos que documentam as principais decisões tomadas.

2.12.1. Banco de Dados

Os Bancos de dados possuem diferentes estratégias para organizar os dados e facilitar o acesso e manipulação. O banco de dados possibilitará manter as informações inseridas pelos usuários através do sistema a qual farão uso, com a base dados contendo informações é possível utilizá-las de acordo com a necessidade apresentada durante as etapas anteriores

(Levantamento de Requisitos e Analise). LAUDON & LAUDON (1999, pg. 4) define que banco de dados é:

Uma coleção de dados organizados de tal forma que possam acessados e utilizados por muitas aplicações diferentes. Em lugar de armazenar dados em arquivos separados para cada aplicação, os dados são armazenados fisicamente de um modo tal que aparentam aos usuários estar armazenado em um só local.

Para armazenar informação em um banco de dados utilizam-se tabelas dentro do banco, onde são divididas e organizadas de acordo com as necessidades de apresentá-las na sua forma mais simples. Com estas tabela é possível organizar as informações inseridas pelos usuários, possibilitando analisá-las e manipulá-las de acordo com a necessidade. Os bancos de dados mais populares de hoje são os bancos relacionais.

DATE (2000) afirma que um sistema de banco de dados é basicamente um sistema computadorizado de armazenamento de registros; isto é, um sistema computadorizado cujo propósito geral é armazenar informações e permitir ao usuário buscar e atualizar essas informações quando solicitado.

Na figura 14 é apresentada uma simplificação de um sistema de banco de dados, que mostra os quatro componentes principais: dados, hardware, *software* e usuário.

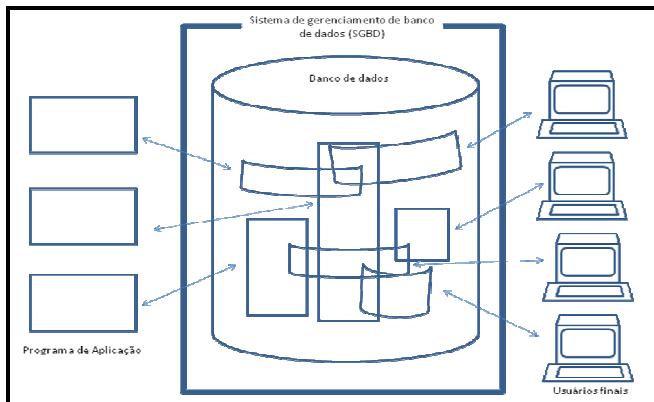


Figura 15: Representação simplificada de um sistema de banco de dados.
Fonte: DATE (2000).

2.12.2. Sistema Gerenciador de Banco de Dados

Um sistema gerenciador de banco de dados (SGBD) tem a capacidade de fornecer mecanismos para armazenar, organizar, recuperar e modificar dados inseridos pelo usuário. Este sistema ainda terá a capacidade de gerenciar todo o acesso ao banco de dados.

O Sistema Gerenciador de Banco de Dados é um conjunto de programas responsável pela base de dados, onde todas as manipulações das informações feitas pelo usuário através da interface do sistema acontecem.

Um conceito encontrado sobre o processo que manipula as informações através de interface é o de DATE (2000, pg. 07):

Entre o banco de dados físico – isto é, os dados de fato armazenados – e os usuários do sistema há uma camada de *software*, conhecida como o gerenciamento de banco de dados ou servidor de banco de dados ou ainda, com maior freqüência, sistema de gerenciamento de banco de dados.

DATE (2000) destaca que os sistemas de banco de dados deverão ser capazes de lidar com solicitações do usuário para buscar, atualizar ou excluir dados existentes no banco de dados, e usado também para acrescentar novos dados no banco. Conforme São várias as razões pela qual os desenvolvedores utilizam SGBD:

- Recuperação de desastres: o banco de dados é protegido contra quebras de hardware, falhas de discos e alguns erros do usuário;
- Compartilhamento entre usuário: vários usuários podem ter acesso ao banco de dados ao mesmo tempo;
- Compartilhamento entre aplicações: Múltiplos programas de aplicação podem ler e gravar dados no mesmo banco de dados. O banco de dados é um meio neutro que facilita a comunicação entre programas independentes;
- Segurança: Os dados podem ser protegidos contra leitura e gravação não-autorizada;
- Integridade: Podem-se especificar regras que os dados devem satisfazer, um SGBD pode controlar a qualidade dos seus dados além das funcionalidades que podem ser oferecidas pelos programas de aplicação;
- Extensibilidade: Pode-se acrescentar dados ao banco de dados sem afetar os programas existentes. Os dados podem ser reorganizados para melhor desempenho;
- Distribuição de dados: O banco de dados pode ser subdividido por várias instalações, organizações e plataformas de hardware.

Além disso, o SGBD deve monitorar requisições de usuários e rejeitar toda tentativa de violar as restrições de segurança e integridade definidas pelo banco de dados. Também é caracterizado pela recuperação dos dados nele armazenados DATE (2000).

O ciclo de vida da maioria das aplicações de banco de dados inclui as seguintes etapas:

- Projetar aplicação;
- Delinear uma arquitetura específica para vincular a aplicação a um banco de dados;
- Selecionar um SGBD específico;
- Projetar o banco de dados. Escrever os códigos de SGBD para estabelecer as estruturas de banco de dados apropriadas;
- Escrever o código de programação para compensar as deficiências do SGBD, fornecer uma interface para o usuário, validar os dados e executar os cálculos;
- Carregar o banco de dados com informações;
- Executar as aplicações. As consultas e atualizações do banco de dados serão feitas quando necessárias.

O projeto cuidadoso de *software* antes da codificação melhora a qualidade e reduz os custos. Um projeto de banco de dados muitas vezes é chamado de modelo de dados ou esquema.

Existem duas abordagens para o projeto do banco de dados conforme KORTH (1989). A primeira é baseada em atributos, onde compila uma lista de atributos relevantes para a aplicação e sintetiza grupos de atributos que preservam as dependências funcionais. A segunda são as entidades, onde é preciso descobrir as entidades significativas para gerar a aplicação.

A arquitetura de três esquemas, conforme demonstrado na figura 15 demonstra a arquitetura padrão para uma família de aplicações relacionais com banco de dados.

A idéia básica é que um projeto de banco de dados deve compreender três camadas:

- Esquema externo: onde é um projeto de banco de dados sob a perspectiva de uma única aplicação;
- Conceitual: é um projeto de banco de dados na perspectiva da empresa, integra aplicações relacionadas e oculta as peculiaridades básicas do SGBS;
- Interno: trata das limitações e característica de um SGBD específica.

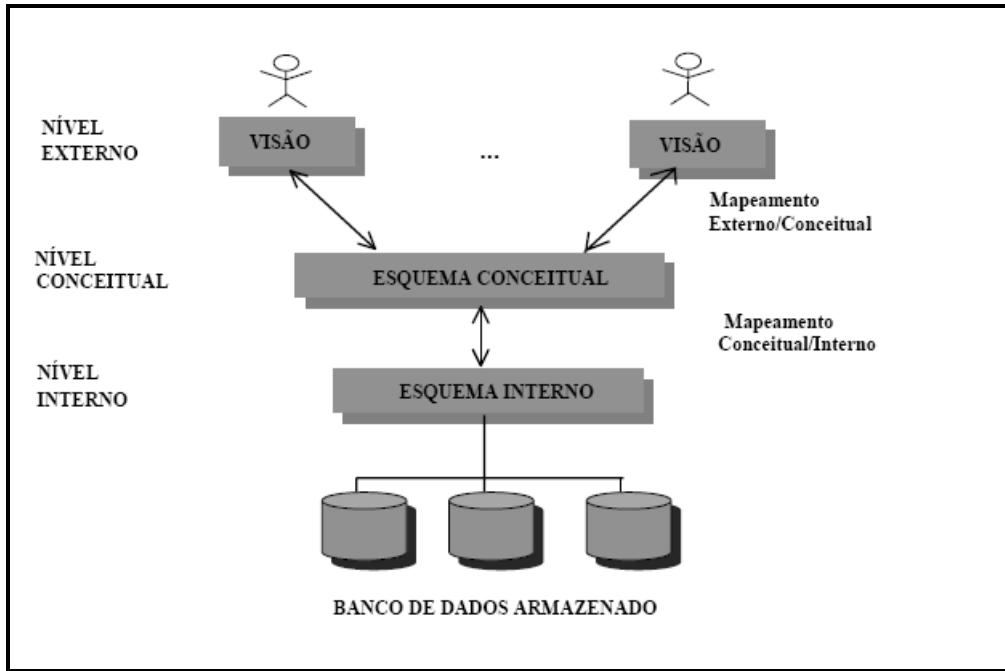


Figura 16: ANSI/SPARC arquitetura de três esquemas.
Fonte: LIA (2008).

2.12.3. Banco de Dados Relacional

Um banco de dados relacional é utilizado para acessar os dados, conforme DEITEL (2006, pg. 897) “banco de dados relacional é uma representação lógica de dados que permite o acesso aos dados sem considerar sua estrutura física”.

Conforme DATE (2000, pg. 741):

A ciência da computação viu muitas gerações de produtos de gerenciamento de dados, começando com arquivos indexados e, mais tarde, SGBDs de rede e hierárquicos...[e] mais recentemente os SGBDs relacionais...Agora, estamos prestes a ver outra geração de sistemas de bancos de dados...[que] oferece o gerenciamento de objetos, [que admite] tipo de dados muito mais complexos.

Conforme DATE (2000), durante o decorrer da evolução, os sistemas relacionais substituíram os antigos sistemas hierárquicos e de rede, tendo em vista está idéia claramente percebemos que os sistemas de objetos irão substituir por sua vez os sistemas relacionais.

Hoje isto já ocorre, com a UML é possível modelar o banco de dados através dos objetos, utilizando a UML como ferramenta. A vantagem da orientação a objeto para modelar o banco de dados, é que se torna um suporte apropriado para tipos de dados e heranças, muito vistas e usadas pelos desenvolvedores.

LEÃO (2002) relata que em junho de 1970, um pesquisador da IBM, chamado E.F. Codd, criou um modelo de Dados Relacional, gerando a base da moderna tecnologia dos databases relacionais. Na forma mais simples, um banco de dados relacional é um simples conjunto de tabelas. Possuem números específicos de colunas e números arbitrários de linhas. As colunas são denominadas atributos e correspondem diretamente aos atributos dos modelos de objetos. As linhas são denominadas tuplas e correspondem as instâncias dos objetos e ligações. Onde um único valor é armazenado em cada interseção de coluna e linha. A integridade de entidades é mantida com as chaves primárias e a integridade referencial com as chaves estrangeiras. Seu nascimento conforme LEÃO (2002) deu-se devido à necessidade de executar operações relacionais como diferença, divisão, intersecção, junção, produto cartesiano, projeção e união.

Conforme MULLER (2002) a teoria sobre banco de dados relacionais afirma que para cada atributo deve ser atribuído um domínio. Um domínio é um conjunto de valores validos.

Acredita-se que o banco de dados relacional através de domínios é capaz de lidar com todo tipo de dados “problemáticos” que com freqüência se afirma que sistemas de objetos podem tratar e sistemas relacionais não podem, como: dados de séries temporais, dados biológicos, dados financeiros, dados de projeto de engenharia, dados de automação de escritórios, e assim por diante.

2.12.4. Modelagem de Dados

A modelagem do banco de dados inicia a partir das informações analisadas após o levantamento de requisitos, onde são identificados os objetos e tabelas que farão parte do sistema. Para se desenvolver um banco de dados é preciso preparar os dados contidos nas tabelas, a forma na qual é feita esta preparação é chamada de normalização.

Segundo MULLER (2002, pg. 3):

A modelagem ajuda você a organizar sua forma de pensar sobre os dados, esclarecendo o significado e a aplicação prática deles. Ajuda-o não somente a comunicar as necessidades, mas também a comunicar de que forma pretende atende-las. Provê uma plataforma a partir da qual você pode utilizar para projetar e construir certo grau de segurança quanto ao sucesso.

A modelagem de dados é uma das primeiras etapas em um projeto de banco de dados. Com o modelo de dados reduz a complexidade a um nível que possibilite o projetista a compreender e manipular o banco.

À medida que os bancos de dados e as estruturas de dados se tornam cada vez mais numerosas e complexas, a modelagem de dados assume uma importância crescente. Conforme MULLER (2002, pg. 4) “Por um lado, sua contribuição advém da sua capacidade de extrair a essência do sistema da obscuridade das estruturas físicas e conceituais e, por outro lado, da multiplicidade de usos. É por tanto, capaz de revelar a essência do sistema”.

A modelagem de dados permite identificar domínios, chaves candidatas e chaves estrangeiras, podendo assim facilitar o entendimento do modelo. Após esta identificação é preciso organizar os dados, para apresentar de uma forma clara e precisa, é nesta etapa que é preciso utilizar a normalização, onde é feita através da análise dos dados que compõem as estruturas utilizando as formas normais, que são conjuntos de restrições às quais dados devem satisfazer e serve para analisar tabelas e organizá-las de forma que sua estrutura seja simples.

Para normalização completa dos dados é feita seguindo as três formas normais existentes:

- Primeira forma normal: Conforme DATE (2000, pg. 306) “uma variável de relação está em 1FN se e somente se, em todo valor valido dessa variável de relação, cada tupla (linhas) contém exatamente um valor para cada atributo”. Nesta forma normal são retirados os atributos de domínio multivalorados da tabela e usa-se o atributo chave para refazer a ligação e recuperação do conteúdo da tabela original.
- Segunda forma normal: “supondo apenas uma chave candidata, que consideremos chave primaria: uma variável de relação está em 2FN se e somente se ela está em 1FN e todo atributo não-chave é irredutivelmente dependente da chave primária ”. Nesta segunda forma são retirados os atributos que dependem funcionalmente de parte da chave da tabela o que seriam chamadas chaves compostas.
- Terceira forma normal: DATE (2000) afirma uma variável de relação está em terceira forma normal (3FN) se e somente se os atributos não chaves são: mutuamente independentes e irredutivelmente dependentes da chave primaria. Na 3FN retira os campos que são dependentes de campos que não são chaves da tabela.

Para esclarecer melhor, atributo não chave é qualquer atributo que não participa da chave primária da variável de relação considerada. Dois ou mais atributos são mutuamente independentes se nenhum deles é funcionalmente dependente de qualquer combinação dos

outros. Esta independência implica que cada um desses atributos pode ser atualizado independentemente dos demais (DATE, 2000).

Segundo MACHADO (2001, pg. 53), “a normalização consiste em definir o formato lógico adequado para as estruturas de dados das tabelas de um banco de dados relacional, com o objetivo de minimizar o espaço utilizado pelos dados e garantir a integridade e confiabilidade das informações”.

Com a normalização é possível eliminar redundâncias e inconsistências de um banco de dados com reorganização dos dados, tornando o banco mais consistente e seguro. Os dados são representados por meio de linhas em tabelas, e essas linhas podem ser interpretadas diretamente como proposições verdadeiras. São fornecidos operadores para operações sobre linhas em tabelas e esses operadores admitem de forma direta o processo de dedução de proposições verdadeiras adicionais a partir das proposições dadas.

2.12.5. MER – Modelo Entidade Relacionamento

O MER é a visão dos dados com alto nível de abstração, ou seja, voltado para o mundo real e sem preocupar-se com detalhes do ambiente operacional, buscando representar um sistema através das chamadas entidades e seus relacionamentos. Segundo SILBERSCHATZ Et al(1999) diz que o Modelo Entidade Relacionamento MER por base a percepção de que o mundo real é formado por um conjunto de objetos chamados entidades e pelos conjuntos dos relacionamentos entre esses objetos.

A principal vantagem do MER é a simplicidade. O Modelo de Entidades e Relacionamentos possui apenas três componentes básicos: Entidade, Atributo e Relacionamento (e seus respectivos símbolos para diagramação).

Conforme SILBERSCHATZ (1999) entidade é um objeto de dados, que representa genericamente um componente do mundo real, sobre o qual desejamos armazenar informações. As entidades podem representar coisas tangíveis (pessoal, material, patrimônio, etc.) ou intangíveis (eventos, conceitos, planos, etc.). Esses dados que serão armazenadas são os atributos.

O relacionamento é a representação das associações existentes entre entidades no mundo real. Muitos relacionamentos não têm existência física ou conceitual, outros dependem das associações de outras entidades. São representados com uma simples linha conectando duas entidades. A conectividade ou cardinalidade, informa o numero de ocorrências de entidades as

quais uma entidade pode estar associada através de um relacionamento. Podendo haver ocorrências de um para um, ou um para muitos e muitos para muitos.

Para notar graficamente uma entidade emprega-se um retângulo identificado por um substantivo (simples ou composto) conforme a figura 16:

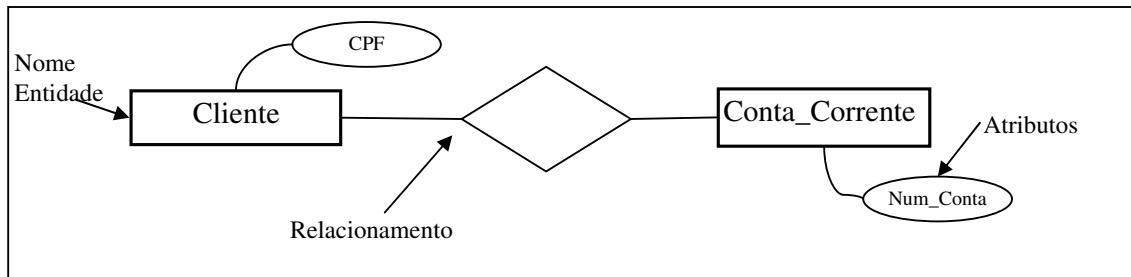


Figura 17: Representação gráfica de Entidade.

Fonte: Guilherme Post adaptado de SILBERSCHATZ (1999).

2.12.6. Dicionário de Dados

Para ajudar a cumprir e documentar as etapas acima mencionadas o SGBD deve fornecer um dicionário de dados, utilizados para a documentação das variáveis usadas para identificar os dados que irão receber do sistema na qual o usuário irá passar as informações. Segundo DATE (2000, pg. 38) “o dicionário de dados pode ser considerado um banco de dados em si (mas de dados do sistema, não um banco de dados do usuário)”. Na tabela 3 a seguir há uma representação de uma tabela utilizada para documentar o dicionário de dados:

Tabela 3: Dicionário de dados.

ID	Campo	Descrição	Tipo do Campo	Tamanho	Obrigatório
Chave	id_cli	Código do cliente	Integer	10	Sim
	cli_nome	Apresenta a descrição do nome do cliente	Char	50	Sim

Fonte: O autor.

2.12.7. SQL (Structured Query Language)

O SQL é uma linguagem relacional. Isto significa que é baseada no modelo de dados relacional primeiramente publicada por E.F. Codd em 1970. Foi entre 1974 e 1975 que a IBM começou a desenvolver a linguagem SQL que inicialmente se chamava SEQUEL (*Structured English Query Language*).

LEÃO (2002) argumenta que o objetivo desta linguagem era fornecer comandos mais intuitivos, mais próximo do idioma falado o inglês, e que muitas das tarefas executadas pelo computador pudessem ficar embutidas nesses comandos, permitindo que os programadores se preocupassem apenas com as rotinas do seu sistema, distanciando-se cada vez mais da máquina. Esta linguagem foi então implantada em um protótipo chamado SEQUEL-XMR.

No período entre 1976 a 1977 houve mais mudanças, ocorreu várias revisões na qual passou a ser chamada de SEQUEL/2. Em 1977, a IBM desenvolve um novo protótipo chamado de System R que utilizava a linguagem SEQUEL/2, já chamada de SQL.

O System R foi bem aceito pelos usuários, devido à sua facilidade de utilização e, sendo assim, a IBM passou a desenvolver a linguagem SQL baseada na tecnologia do System R.

LEÃO (2002) relata que anos mais tarde a IBM e outros fabricantes de *softwares* lançaram seus produtos utilizando a linguagem SQL, como, por exemplo:

- SQL/DB (IBM);
- DB2 (IBM);
- ORACLE (Oracle Corp.);
- DG/SQL (Data General Corp.);
- SYBASE (Sybase Inc.).

A partir de 1982, o American National Standard Institute (ANSI) propôs uma padronização da linguagem relacional SQL. Mas somente em 1986 a proposta foi aceita, esta proposta consistiu essencialmente no dialeto criado pela IBM. Em 1987, o padrão ANSI também foi aceito pelo Instituto Internacional Organization for Standardization (ISSO).

LEÃO (2002) informa que esta versão do SQL foi então informalmente chamada de SQL/86. Anos mais tarde os instituto ISSO e ANSI criaram uma nova definição da linguagem e o produto final passa a ser chamado de SQL2 ou SQL/92. A partir de então, o SQL/3 estava em desenvolvimento. É plano de torná-lo uma linguagem de programação completa e orientada a objeto com recursos como métodos, heranças, polimorfismo, encapsulamento e todos os recursos associados à orientação a objeto. As linguagem para processamento em banco de dados são chamadas de “Linguagem de Quarta Geração”.

A linguagem SQL é relativamente simples de ser utilizada, pois possui poucos comandos. Cada um desses comandos possui várias cláusulas dentro do padrão ANSI/ISSO, e

cláusulas próprias específicas de cada produtor de *software* que tem com o objetivo diferenciar seu produto no mercado de *softwares* para banco de dados. Com a possibilidade de criar essas cláusulas torna a sintaxe mais complexa, mas mesmo assim a facilidade de seu uso é enorme se comparada as dificuldades encontradas nas linguagens procedurais.

2.13. Implementação

Esta fase é responsável pela codificação do sistema que foi organizada durante as etapas anteriores. Neste momento é desenvolvida na prática a implementação do código apresenta na documentação antes vista (Levantamento de Requisitos e Analise). Onde a visão desta etapa nos artefatos físicos (programas, bibliotecas, banco de dados) utilizados na efetiva montagem do sistema. Criando rotinas e organizando o sistema de acordo com as regras de negócios acordadas e compreendidas pelo engenheiro de *software* ou analista e usuários. Segundo LAUDON & LAUDON (1999, pg. 245):

É aqui, no fornecimento das instruções reais a máquina, que o coração do sistema toma forma. Durante a fase de programação, as especificações do sistema que foram preparadas durante a fase de projeto são traduzidas em código de programa.

Durante a etapa de implementação, é preciso se desenvolver a codificação do *software* dentro dos padrões de cada linguagem, para assim poder alcançar a qualidade dos códigos desenvolvidos, mantendo uma endentação correta para facilitar a leitura, manutenibilidade, padrões de escrita de código, nomenclaturas de variáveis e objetos.

Esses objetivos são alcançados desde que a equipe de desenvolvimento faça uso de uma linguagem de programação que dominem, caso houver a necessidade de se desenvolver em outras linguagens que não é de conhecimento da equipe, é preciso preparar um treinamento para que todos possam ter a oportunidade de apreender, e sabendo que isto irá influenciar no prazo de entrega do projeto.

2.13.1. Linguagens de Programação

As linguagens de programação são ferramentas utilizadas na codificação dos *softwares*, são métodos e padrões pré-definidos para expressar através de códigos as funcionalidades de um sistema. Conforme DEITEL (2006, pg. 5) “Os programadores

escrevem instruções em várias linguagens de programação, algumas diretamente compreensivas por computadores e, outras, requerendo passos intermediários de tradução”.

A primeira geração de linguagens contempla uma codificação em linguagem de máquina: que é interpretada por qualquer computador. É uma linguagem natural definida pelo seu projeto de hardware. Possui Strings que instruem os computadores a realizar suas operações mais elementares uma de cada vez. Normalmente são caracterizadas por serem muito complexas para os seres humanos. Exemplo: DEITEL (2006, pg. 5) soma em horas extras ao salário-base e armazena o resultado no salário bruto (usando linguagem de máquina):

+1300042774

+1400593419

+1200274027

A linguagem *assemblers* representa a primeira geração de linguagem de programação. Essas linguagens dependentes da máquina exibem o mais baixo nível de abstração com o qual um programa pode ser representado. A linguagem *assembly* possui programas tradutores *assemblers* que foi desenvolvido para converter os primeiros programas de linguagem *assembly* em linguagem de máquina a velocidade de computador. Sua característica se deu por ter um código mais limpo e claro, mas sendo ainda um a linguagem que os programadores precisavam utilizar de muitos recursos de instruções para realizar pequenas tarefas. Exemplo: DEITEL (2006, pg. 5) soma em horas extras ao salário-base e armazena o resultado no salário bruto (usando linguagem *assembly*):

Load	basepay
Add	overpay
Store	grosspay

Na década 1950 foram desenvolvidas as linguagens de segunda geração e no começo da década de 1960 surgem todas as linguagens de programação moderna (terceira geração). As linguagens de segunda geração são caracterizadas pelo amplo uso, enormes bibliotecas de *software* possuem a programação multi-usuário, sistema este com execução em tempo real e a criação dos gerenciadores de banco de dados e a mais ampla familiaridade e aceitação. As

linguagens que ficaram conhecidas nessa época foram o *Fortran*, *Cobol*, *Algol* e algumas extensões como *Basic*.

Fortran é uma linguagem que tem resistido a 30 anos de críticas segundo PRESSMAN (1995), mas é ainda muito utilizada na área de engenharia e pela comunidade científica. O *Fortran* para aplicações que lidam com números continua sendo a linguagem mais preferida, mas para aplicações em *software* básico, de tempo real , outras linguagens oferecem vantagens irresistíveis.

Cobol é uma linguagem que foi aceita e ainda continua em uso, com menor freqüência, para aplicações comerciais.

Segundo PRESSMAN (1995) “*Algol* foi o precursor de muitas linguagens de terceira geração, por oferecer poderosas estruturas de controle e tipos de dados. O *Basic* foi uma linguagem originalmente criada para o aprendizado e teve seu uso bastante reduzido já na década de 1970”.

A terceira geração surgida na metade da década de 70 foi também chamada linguagem moderna ou estruturada. Essa linguagem foi considerada mais inteligente e também não exigia um *hardware* muito robusto. LAUDON & LAUDON (1999, pg. 135), citam que “os programas se tornaram fáceis de criar e começaram a ser usados mais amplamente para problemas científicos e empresariais”.

Linguagem de alto nível: são instruções únicas que podem ser escritas para realizar tarefas substanciais. Possui programas tradutores conhecido como compiladores, utilizados para converter os programas de linguagem de alto nível em linguagem de máquina. Sua característica surgiu devido à forma com que se escreve o código ser parecido com o inglês cotidiano, tendo notações matemáticas como as outras. Exemplo: DEITEL (2006, pg. 5) soma em horas extras ao salário-base e armazena o resultado no salário bruto (usando linguagem assembly):

```
grossPay = basePay + overPay
```

As linguagens de alto nível são mais utilizadas pelos programadores, por possuírem um ambiente de programação e uma linha de código mais clara, com muitos recursos e fácil de interpretar. Algumas das linguagens de alto nível são: Visual Basic, C, C++, NET, Visual C++.NET e Java.

Com todas estas linguagens a característica que mais se busca em uma linguagem de programação é sua legibilidade, com isto o programador consegue entender através de um trecho de código qual a seqüência das rotinas que estão sendo executadas.

2.13.1.1. Linguagem Java

Esta linguagem de programação foi desenvolvida para utilização em pequenos dispositivos eletrônicos inteligentes. Foi criada pela Sun, e teve seu alge em 1993 quando houve o surgimento da internet, que trouxe várias oportunidades para as empresas desenvolvedoras de softwares.

Em 1995, a Sun anunciou o Java, como uma nova plataforma de desenvolvimento. No inicio o Java foi utilizado para elaboração de páginas para World Wide Web, produzindo conteúdos interativo e dinâmico, utilizando applets com imagens em movimento.

Desde 1996 a linguagem Java vem crescendo produzindo soluções desde pequenas aplicações até aplicativos corporativos. Java também invadio os celulares, PDAs assim como outros dispositivos eletrônicos de pequeno porte.

FURGERI (2002, pg. 18) cita ainda que:

Praticamente todos os principais fabricantes de software sentiram a necessidade de lançar no mercado alguma ferramenta para manipular o Java, o que mostra sua força e longevidade para os próximos anos no ambiente das linguagens de programação mais usadas.

A Microsoft tentou desenvolver o Visual J++ como uma de suas ferramentas de desenvolvimento, mas sem muito sucesso, e acabou desaparecendo da nova versão do Microsoft Studio depois de sofrer alguns processos pela Sun.

Conforme FURGERI (2002) o Java têm feito muito sucesso devido ao fato de que os programas escritos em Java podem ser executados virtualmente em qualquer plataforma, aceitos em qualquer tipo de computador (ou outros aparelhos).

Uma característica forte desta linguagem é o suporte a multiplataformas, que facilita muito o trabalho dos programadores que não precisam estar preocupados com em qual máquina o programa será executado.

O Java possui outras características como: uma linguagem orientada a objeto, possui uma ótima portabilidade, multithreading, suporte a comunicação em rede e acesso remoto a banco de dados.

Os programas desenvolvidos em Java funcionam como acessórios (chamados de applets) onde são colocados no computador do usuário no momento que ele acessa um site qualquer, onde o computador do usuário começa a executar um programa armazenado no servidor web que é transferido para sua maquina no momento do acesso.

Em meados da década de 1980 foram desenvolvidas as ferramentas de quarta geração, que tem como características principais a geração de sistemas especialistas, o desenvolvimento de inteligência artificial e a possibilidade de execução dos programas em paralelo.

O intuito destas linguagem é permitir ao usuário desenvolver programas complexos e como cita LAUDON & LAUDON (1999, pg. 135), “essas linguagens reduziram drasticamente o tempo de programação e tornaram as tarefas do software tão fáceis que muitas podiam ser executadas por usuários comuns de computadores, sem a ajuda de programadores profissionais”.

Com a quarta geração, surgiu uma linguagem de consulta muito utilizada, o SQL, usada para consulta e manipulação em banco de dados visto a seguir.

2.13.2. Integração

Dentro das organizações existe por uma razão natural a troca de informação entre sistemas que estão em diversos departamentos, pois somente assim é possível identificar e gerenciar uma organização. A integração é um fator importante que depende muito do entendimento de todos os envolvidos na organização.

Conforme REZENDE (2005, pg. 101):

As integrações dos sistemas de informação são as relações de interdependências entre sistemas ou subsistemas existentes na organização e o seu meio ambiente interno e externo, que dinamizam a troca de dados e informação entre eles. Não é mais possível aceitar que os sistema sejam isolados e completamente independentes nas organizações.

As necessidades de integração de sistemas sempre existem, varias tecnologias possibilitam o intercâmbio de informações, as regras de negócio envolvidas e oferecem maior flexibilidade para o direcionamento das mensagens envidas de um sistema ao outro, como nos departamentos, toda organização precisa fazer com que a comunicação flua de uma maneira amena para que os departamentos que dependem de outros possam receber as informações mais claras.

2.14. Testes

É durante está etapa que é preciso se realizar e efetuar uma verificação de todas as funcionalidades do sistema, sendo ainda avaliada pela equipe de qualidade, onde irá garantir que o software tenha sido desenvolvido de acordo com os requisitos do cliente.

MARTINS (2002) define que “as principais etapas a serem concluídas na fase de testes são: fazer os planejamentos dos testes para cada interação, projetar o processo de teste criando casos de testes, rotinas de testes, executar os testes planejados e documentar todos os resultados”.

Este processo é demorado, mas precisa ser executado com o máximo de atenção pela equipe da qualidade no desenvolvimento de sistemas. As funções do sistema, assim como relatórios, janelas, eventos precisam ser vistas com o maior detalhe possível, é preciso tentar achar erros, para fazer as devidas correções ainda antes de ser entregue ao cliente. Durante a fase de testes é possível realizar duas ações, a de verificação que se refere ao conjunto de atividades que garantem que o software implementa corretamente uma função específica, e a de validação refere a um conjunto de atividades diferentes que garantem que o software construído corresponde aos requisitos do cliente.

BEZERRA (2002, pg. 26) conclui ainda que “o principal produto dessa fase é o relatório de testes, contendo informações sobre erros detectados no software. Então, após a atividade de testes, os diversos módulos do sistema são integrados, resultando finalmente no produto de software”.

Com base nos conceitos apresentados percebe-se a importância de se realizar os testes em todas as etapas do desenvolvimento do software, podendo ser iniciada o quanto antes, assim que forem disponibilizados os primeiros subsistemas, podendo assim diminuir o risco de encontrar muita inconsistência ao final do projeto. Os principais testes são de unidade que consiste na verificação da menor unidade de um projeto de sistema, teste de integração que é o teste em conjunto utilizando demais módulos que compõem o sistema e o teste de sistemas que consiste em unir todos os elementos que compõem o sistema (hardware, pessoal, informação, software) e realizar verificações. Os testes são importantes para se desenvolver um sistema confiável e eficiente. Os tipos de testes utilizados para validar os sistemas são visto a seguir.

- **Testes de Funcionalidade:** Esta categoria de testes tem por objetivos simular todos os cenários de negocio e garantir que todos os requisitos funcionais sejam

implementados. Conforme BERTIÉ (2002) os testes funcionais exigem profundo conhecimento das regras de negocio de uma aplicação para que todas as variações possíveis sejam simuladas, obtendo o máximo de cobertura dos cenários de negócios.

- **Teste de Usabilidade:** Essa categoria de teste tem por objetivo simular as condições de utilização dos software sobre a perspectivas do usuário final. Segundo BARTÉ (2002) a idéia desses teste é medir o nível de facilidade disponibilizada pela aplicação, de modo deixar o software mai simples e intuitivo. Dessa forma, esses testes focalizam a facilidade de navegação entre as telas da aplicação, a clareza de textos e mensagens que são apresentados ao usuário, o acesso simplificado de mecanismos de apoio ao usuário, o volume reduzido de interações para realizar uma determinada função, padronização visual, entre outros aspectos.
- **Teste de carga:** Tem por objetivo simular condições atípicas de utilizações do software, provocando aumentos e reduções sucessivas de transações que superem os volumes máximos previstos para o software, gerando continuas situações de pico e avaliando o software e todas as infra-estruturas estão se comportando. Conforme BERTIÉ (2002) a idéia é avaliar como todo o conjunto da solução lida com variações sucessivas de processamento.
- **Testes de configuração:** Tem por objetivo executar o software sobre diversas configurações de software e hardwares. Conforme BERTIÉ (2002) a idéia é garantir que a solução tecnológica “rode”adequadamente sobre os mais variados ambientes de produção previstos nas fases de levantamento dos requisitos.
- **Testes de Performance:** O objetivo principal é determinar o desempenho, nas situações previstas de pico máximo de acesso e concorrência, está consistente com os requisitos definidos. BARTIÉ (2002) define que o critério de sucesso aqui estabelecido é empregar o volume de transações e tempo de resposta obtidos nos testes e compará-los com os valores-limite especificados.
- **Testes de Instalação:** Objetivo é validar os procedimentos de instalações de uma aplicação, bem como avaliar se estes possilitam as varias alternativas previstas nos requisitos identificados. BERTIÉ (2002) destaca que a idéia principal é aplicar as

muitas variações de instalação (normal e alternativa) e avaliar seu comportamento durante esses procedimentos.

- **Teste de segurança:** Tem como objetivo destacar as falhas de segurança que podem comprometer o sigilo e a fidelidade das informações, bem como provocar a perda de dados ou interrupções de processamento. Esses ataques à segurança do software podem ter origens internas ou externas, provenientes de hakers, quadrilha especializada, profissionais descontentes ou mesmo pessoas com intenção de ganhos ilícitos BERTIÉ (2002).
- **Teste de Confiabilidade e Disponibilidade:** Essa categoria visa monitorar o software por um determinado período de tempo e avaliar o nível de confiabilidade da arquitetura da solução. Essas informações devem ser coletadas durante a execução dos próprios testes de sistema, identificando sempre quando uma interrupção foi produzida por uma falha da infra-estrutura (confiabilidade) e contabilizando o tempo necessário para resolução desse problema (disponibilidade). BERTIÉ (2002) diz que os testes listados a seguir são importantes juntamente com os teste aceite, pois o ambiente fica à disposição do cliente.
- **Teste de Recuperação:** Conforme BERTIÉ (2002), o objetivo desta categoria é avaliar o comportamento do software após a ocorrência de uma erro ou de determinadas condições anormais. Algumas aplicações suportam, solução de missão crítica, exigindo alto índice de disponibilidade do software. Nesse caso, a arquitetura da aplicação deve ser tolerante a falhas, ou seja, no caso de erros de qualquer natureza, o software deve ter a capacidade de se manter em execução até que a condição de impedimento desapareça.
- **Teste de Contingência:** Essa categoria de teste visa validar os procedimentos de contingência a ser aplicada à determinada situação prevista no planejamento do software. A idéia é simular os cenários de contingência e avaliar a precisão dos procedimentos. Esse teste deve ser realizado pela própria equipe de plantão, na qual o tempo total de execução do plano de contingência também será avaliado.

2.15. Implantação

A implantação do Software é a entrega do produto ou a versão final ao cliente, normalmente é feito toda uma preparação para realizar a entrega do produto de software.

Onde são disponibilizados os consultores que estarão se deslocando até a empresa para acompanhar todos os processos, podendo efetuar treinamentos com os usuários, para não criar desconforto com o novo software e gerar rejeições. Segundo LAUDON & LAUDON (2001, pg. 265) "o estágio de instalação consiste nos passos finais para colocar o novo ou modificado sistema em operação: teste, treinamento e conversão".

2.16. Manutenção de Sistemas

A manutenção do software segundo alguns autores pode representar entre 40 a 60% do custo do projeto. Conforme GUEDES (2004, pg. 23) “a modelagem é necessária para diminuir custos com a manutenção, se a modelagem estiver correta o sistema não apresentará erros e então não precisará de manutenção”.

A manutenção de software em muitos casos é inevitável, devido a complexidade do sistema, ou em muitas vezes no caso de que as empresas são dinâmicas e elas mudam constantemente, onde surgiram novas necessidades que não existiam no projeto inicial, isso sem falar nas freqüentes mudanças de leis, alíquota, impostos, taxas ou no formato de notas fiscais assim como vemos hoje, o governo decretou que as empresas são obrigadas a emitir a nota fiscal eletrônica.

A manutenção se torna mais fácil quando o projeto do desenvolvimento do software seguiu uma boa modelagem, uma ótima documentação, dentro dos padrões de linguagens, por que o profissional que vai prestar manutenção nem sempre é o mesmo que desenvolveu. Conforme GUEDES (2004, pg. 23) define manutenção:

É uma tarefa ingrata pelos profissionais de desenvolvimento, por normalmente exigir que estes despendam grandes esforços para compreender códigos escritos por outros profissionais com estilos de desenvolvimento diferentes e que normalmente não se encontram mais na empresa.

Estes códigos segundo autor Guedes são “códigos alienígenas”, onde se refere a códigos que não seguem regras atuais de desenvolvimento da empresa, não houve uma modelagem, pouco ou nenhuma documentação. Deixando assim o programador sem amparo,

levando tempo para corrigir erros que em muitas vezes seriam resolvidos rápido se tivesse uma documentação do sistema.

Com a modelagem e a documentação correta se torna mais rápido e viável a manutenção, sem ter que estressar toda a equipe de desenvolvimento.

A manutenção normalmente acontece após o inicio de seu uso pelos clientes, sendo algumas vezes feita para corrigir problemas encontrados pelo usuário durante seu uso e outras vezes para a implementação de novas funcionalidades.

Segundo LAUDON & LAUDON (1999, pg. 247) é que “são chamadas manutenção as mudanças em hardware, software, documentação ou procedimentos em um sistema de produção para corrigir erros, atender novas exigências ou melhorar a eficiência do processamento”.

Mesmo após houver ocorrido as devidas manutenções solicitadas pelo cliente, é preciso ser documentado e modelado as alterações, informando a versão, e as pessoas envolvidas na alteração para não desatualizar a documentação do sistema e prejudicar futuras manutenções. GUEDES (2004) defende que já que muitas vezes uma documentação desatualizada pode ser mais prejudicial à manutenção do sistema do que nenhuma documentação. Para evitar este tipo de problema é preciso seguir padrões, que são estipulados pelas organizações responsáveis pelo desenvolvimento de software.

2.17. Processos Operacionais

Os processos operacionais são a forma com que a empresa desenvolve suas atividades diárias. Os sistemas de informação têm a capacidade de controlar esses processos fornecendo informações geradas pelas atividades desenvolvidas de uma forma clara que possibilite os responsáveis pela empresa a tomarem decisões.

2.17.1. Vendas

Este processo é responsável por manter as vendas de uma organização. É durante este processo que a empresa absolve um maior numero de informações, que se gerenciadas por um sistema de informação é possível traçar estratégias e tomar decisões em cima das informações apresentadas.

Assim como em outras áreas, um sistema de informação tem a capacidade de gerenciar todas as informações geradas por este processo, auxiliando a organização na diminuição de custos e mostrando tendências que o mercado está seguindo, aumentando assim o crescimento da empresa gerando lucro e tornando-a mais competitiva no mercado.

Com o avanço contínuo da venda pela internet TURBAN (2003) relata que a alternativa de comprar em casa pressiona o varejista a oferecerem mais produtos e um único local e a fornecer um serviço melhor. O aumento do número de produtos e o desejo dos clientes no sentido de obter mais informações dentro da loja geram uma necessidade de se adquirir um sistema que seja de fácil manuseio mas de grande eficácia, para o vendedor realizar a venda sem ter quer muitas vezes realizar uma grande quantidade de processos para vender apenas cinco balas sem perder outros clientes que exigem um atendimento mais rápido.

Os objetivos de controlar o processo de vendas é fazer com a organização além de diminuir custos com papéis, fichários e outros materiais utilizados na forma tradicional do processo de vendas de uma empresa, é poder gerenciar o que se vende, agilizando e tornando seguras as informações geradas por este setor. SILVA (1990) diz que, “os controles são dos mais variados: numeração e remessa de pedidos, relatório de vendas e o cumprimento das normas administrativas da empresa”. Podendo ainda com as informações obtidas neste processo ser utilizada em outros departamentos, como contas a receber, contas a pagar e controle de estoque.

2.17.2. Contas a Receber

Este processo existe pela necessidade de auxiliar o processo de vendas. Podendo ficar única e exclusivamente voltado para gerenciar o quanto se vende de quem está se recebendo, quando dará entrada tal recebimento, além de auxiliar o processo da venda. Agindo automaticamente, assim que um cliente solicita fazer um pedido de venda, o sistema lança as informações em uma tabela que armazenará as informações das vendas no banco de dados.

Conforme STAIR (1998) uma aplicação importante de contas a receber é identificar os riscos de mau crédito. Uma outra função é monitorar o tempo de débito passados e tentar minimizar o tempo das contas não-pagadas”.

CONSO (1972) afirma que administração de vendas é a parte essencial de um sistema de gestão, uma vez que ela abrange completamente as operações que lhe imprimem o ritmo de funcionamento.

Com isso a empresa consegue diminuir e identificar os clientes inadimplentes, reduzindo custos, gerenciando recebimentos, e apresentando dados que auxiliaram os gerentes a tomarem decisões importantes para o bom desempenho do setor.

2.17.3. Contas a Pagar

Este processo inicia-se assim que é finalizada uma solicitação de compra a um determinado fornecedor. Onde são lançadas as informações de valores, datas de vencimento, forma e condição de pagamento solicitada ao fornecedor.

Com o gerenciamento deste processo é fácil saber quais são os maiores fornecedores (parceiros), como tem sido feitos os pagamentos ao fornecedor, se foi em cheque, duplicata, qual o valor total comprado.

Fazendo assim com que aumente o relacionamento entre fornecedor e cliente, podendo abrir novas portas através das negociações feitas entre gerentes, que com os dados em mãos poderão brigar por preços e prazos.

Com o controle do processo de contas a pagar e contas a receber, conforme CONSO (1972) as informações obtidas através desses processos servirão de base para os trabalhos contábeis e financeiros.

2.17.4. Compras

Um sistema de informação que gerencie este processo possibilita a partir das informações obtidas nos processos anteriores (vendas, contas a pagar e receber), a saber, qual produto deve ser comprado, de qual fornecedor é possível conseguir preço e prazo e qual a quantidade a ser adquirida.

SLACK (2006) informa que a parte principal do processo de compras não é só registrar a compra mais sim é utilizar as informações das compras feitas aos fornecedores para posteriormente serem avaliadas para identificar se os produtos comprados de tal fornecedor estão com bom preço, ótima qualidade, entrega dentro do prazo.

Com isso é possível realizar uma análise dos dados através de um responsável por realizar as compras para empresa, possibilitando-o a tomar uma decisão que em muitas vezes poderá ser a mais enérgica, que é procurar pesquisar melhores ofertas através de outros fornecedores.

Com o gerenciamento do processo de compra afirma TURBAN (2003) que com o emprego de sistema de informação para gerir os processos conseguimos administrar nossos custos, e tranquilizar os clientes, sabendo que não irão perder viagem ao se deslocarem até a empresa.

Dessa forma é possível diminuir não só custos com a compra de produtos que já possuem em estoque (desperdício), mas evitar a falta de espaço que em muitas vezes se torna um problema para as empresas, saber em qual produto deve ser feito o investimento possibilita um maior controle do que é comprado dos fornecedores.

2.17.5. Controle de Estoque

Este processo é tão importante quantos os outros, é aqui que ocorre a entrada dos produtos solicitados ao fornecedor. Por esta razão se torna necessário o uso de um sistema para gerenciar este processo.

SLACK (2006) afirma que a compra desnecessária de produtos acarreta em uma considerável quantidade de capital parado, assim como os itens mantidos nem estoque podem deteriorar, tornarem-se obsoletos ou apenas perder-se e, além disso, ocupam um espaço valioso no depósito.

O estoque de uma empresa é algo fundamental para manutenção da mesma, se não há produtos em estoque não existe a venda, consequentemente a empresa obtém prejuízos, se tem estoque de produtos de mais, e pouco giro, corre o risco de perder produtos devido a prazos validade, armazenagem inadequadas por falta de espaço. Tendo em vista esses pontos se faz necessário a utilização de recursos para gerenciar a entrada e saída de mercadorias.

Conforme STAIR (1998) para quase todas as empresas, o estoque deve ser rigorosamente controlado. Com um sistema que possibilite gerenciar a entrada e saída de produtos no estoque, a empresa consegue controlar os itens mais ou menos vendidos, qual a quantidade disponível no estoque, através de relatórios e documentos que apontam os gargalos e oportunidades para fazer a compra de produtos certa.

Neste capítulo foram apresentados os conceitos que auxiliaram a fundamentar o relatório, possibilitando através de livros buscar o conhecimento e através das metodologias aprendidas dar inicio a nova etapa que é a Descrição Prática do relatório.

3. DESCRIÇÃO PRÁTICA

Este capítulo é responsável por apresentar um detalhamento completo de todo o desenvolvimento do projeto, com base nos conceitos definidos na fundamentação teórica e as oportunidades oferecidas pelo campo de estágio que se teve através do desenvolvimento de um Sistema para a empresa Agropecuário Selva LTDA ME.

Durante o desenvolvimento deste capítulo será realizado o levantamento de requisitos, análise de requisitos, análise do sistema, *design* e implementação, utilizando como metodologia a UML, conforme mencionada na fundamentação teórica.

3.1. Local de Desenvolvimento do Sistema

Conforme mencionado no capítulo 1, o projeto foi desenvolvido na empresa Agropecuária Selva. A empresa existe há 14 anos atendendo a comunidade Joinvillense, oferecendo ótimos preços, um atendimento de qualidade e uma grande variedade de produtos.

Segundo proprietário este mercado vem crescendo muito e com isso a concorrência aumenta cada vez mais, menciona ainda o quanto é importante controlar os processos da empresa, por esta razão solicitou investir na tecnologia para gerenciar os processos de sua empresa.

Os processos realizados atualmente pela empresa são feitos manualmente, fazendo uso de cadernos, papéis, agendas para fazer as anotações das vendas e das contas a pagar. A empresa não dispõe de um cadastro de clientes, ou de produtos, fica tudo gravado na memória humana dos empregados. Com isto além de desenvolver um sistema para gerenciar os processos é importante realizar o treinamento do uso do sistema, para tornar seu uso mais agradável e de fácil aceitação entre os usuários, possibilitando mostrar a sua capacidade de armazenar dados e apresentá-lo de uma forma legível ao ponto de ser poder ser utilizado nas tomadas de decisões da empresa.

3.2. Sistema atual da empresa

Para controle das contas a pagar, são feitas anotações em agendas, contendo a informação do valor e data de vencimento, onde é feito um controle durante todos os dias, sendo dada a atenção para qual duplicata deve ser paga, ou em que dia deverá ser feito um depósito para cobrir os cheques emitidos ao fornecedor.

Durante a venda é feita anotação somente se a venda for a prazo, onde o cliente leva a mercadoria para pagar em outro dia, estas anotações são feitas em pequenos papeis constando apenas o primeiro nome do cliente, produto, e valor total da compra.

Quando a empresa recebe a entrega de mercadorias solicitadas a um fornecedor, é feita a conferência apenas dos itens que contém na nota fiscal, caso o item solicitado no pedido não conste na nota não é mencionado nem feito abatimento na nota, caso constar na nota e não for entregue é feito o abatimento no ato da entrega.

Os produtos são repostos nas prateleiras sem ser dado entrada, ou anotado em um caderno para se ter controle do que se tem em estoque.

As compras são realizadas sempre que a representante visita a empresa, ou o proprietário identifica a falta de muitas mercadorias, mas não é feito um levantamento do quanto se tem em estoque ou quanto é preciso comprar, normalmente é feito o pedido informando a quantidade comprada pela ultima vez.

3.3. Regras de Negócios

São definidas as regras conforme informações obtidas através das entrevistas realizadas com o cliente, onde é feita uma analise dos processos realizados pela empresa atualmente.

O analista precisar extrair todas as exigências solicitadas pelo cliente é preciso ajustar as regras para que o sistema apresente as características de um software desejável pela empresa, auxiliando a desenvolver um sistema que atenda a necessidades estabelecidas pelo cliente.

Regra de negócio ao emitir venda.

Tabela 4: Regra de negócio – Emitir venda.

Emitir Venda (RN01)

Descrição	Para emitir o pedido de venda será obrigatório: <ul style="list-style-type: none"> ✓ Atendente cadastrado; ✓ Cliente ser cadastrado; ✓ Informar condição de pagamento; ✓ Selecionar forma de pagamento; ✓ Informar ao menos 01 (um) item com embalagem mínima; ✓ Confirmar a venda.
-----------	---

Fonte: Guilherme Post

Regra de negócio ao emitir a venda

Tabela 5: Regra de negócio – Emitir venda PDV.

Emitir Venda PDV (RN02)

Descrição	Para emitir o pedido de venda a partir do PDV (Ponto de Venda) será obrigatório: <ul style="list-style-type: none"> ✓ Informar forma de pagamento; ✓ Inserir ao menos 01 (um) item com embalagem mínima; ✓ Confirmar venda.
-----------	--

Fonte: Guilherme Post

Regra de negócio ao emitir a compra.

Tabela 6: Regra de negócio – Emitir compra.

Emitir Compra (RN03)

Descrição	Para emitir pedido de compra será obrigatório: <ul style="list-style-type: none"> ✓ Comprador cadastrado; ✓ Fornecedor ser cadastrado; ✓ Informar a condição de pagamento; ✓ Selecionar forma de pagamento; ✓ Inserir ao menos 01 (um) item com embalagem mínima; ✓ Confirmar pedido de compra.
-----------	---

Fonte: Guilherme Post

Regra de negócio ao cadastrar cliente.

Tabela 7: Regra de negócio – Cadastrar cliente.

Cadastrar Cliente (RN04)

Descrição	Para cadastrar cliente, será obrigatório: <ul style="list-style-type: none"> ✓ Nome; ✓ Endereço, rua, bairro, número; ✓ Telefone residencial.
-----------	--

Fonte: Guilherme Post

Regra de negócio ao cadastrar fornecedor.

Tabela 8: Regra de negócio – Cadastrar fornecedor.

Cadastrar Fornecedor (RN06)

Descrição	Para se cadastrar Fornecedor, será obrigatório: <ul style="list-style-type: none"> ✓ Razão Social; ✓ CNPJ; ✓ Endereço, rua, número, bairro, cidade; ✓ Inscrição Estadual; ✓ Telefone celular; ✓ Telefone da empresa; ✓ E-mail;
-----------	---

Fonte: Guilherme Post

Regra de negócio ao cadastrar a nota fiscal de entrada.

Tabela 9: Regra de negócio – Cadastrar nota fiscal de entrada.

Cadastrar Nota Fiscal de Entrada (RN7)

Descrição	O usuário conferente durante o recebimento da NF de entrada será responsável por:
	<ul style="list-style-type: none"> ✓ Conferir os itens da nota fiscal com o pedido de compra; ✓ Informar o código do produto que esta sendo dada entrada; ✓ Informar à quantidade que esta recebendo; ✓ Informar o valor unitário do produto; ✓ Informar o valor total da Nota fiscal; ✓ Informar o IPI, ICMS; ✓ Informar o valor do transporte; ✓ Deverá confirmar a forma de pagamento; ✓ Informar a condição de pagamento; ✓ Informar a Data de entrega; ✓ Se houver divergências deverá listar as divergências.

Fonte: Guilherme Post

Regra de negócio ao cadastrar conta.

Tabela 10: Regra de negócio – Cadastrar conta.

Cadastrar Conta (RN8)

Descrição	O gerente cobra a conta para lançar as contas a receber ou a pagar, para isso deverá informar a descrição da conta a ser criada.
-----------	--

Fonte: Guilherme Post

Regra de negócio ao cadastrar funcionário.

Tabela 11: Regra de negócio – Cadastrar funcionário.

Cadastrar Funcionário (RN05)

Descrição	Para cadastrar Funcionário, será obrigatório:
	<ul style="list-style-type: none"> ✓ Nome completo; ✓ Data nascimento; ✓ Endereço, rua, número, bairro, cidade e estado; ✓ Informar o RG, CPF; ✓ Referências ✓ Telefone residencial; ✓ Informar o salário a ser pago; ✓ Data de admissão; ✓ Número carteira de trabalho e serie; ✓ Endereço de e-mail.

Fonte: Guilherme Post

Regra de negócio ao analisar clientes inadimplentes.

Tabela 12: Regra de negócio – Clientes inadimplentes.

Clientes Inadimplentes (RN9)

Descrição	Ao efetuar a venda, atendente confirmará pedido de venda, sistema consultará status do cliente, se houver duplicatas em aberto, cheques devolvidos, sistema mostrará mensagem informando “pedido retido”, será permitida a venda mediante autorização da gerência.
-----------	--

Fonte: Guilherme Post

Regra de negócio ao analisar pedidos retidos.

Tabela 13: Regra de negócio – Pedidos retidos.

Pedidos Retidos (RN10)	
Descrição	A retenção dos pedidos poderá ocorrer quando, a cheques devolvidos, alto índice de atraso, cliente não cadastrado, atendente não cadastrado.

Fonte: Guilherme Post

Regra de negócio aos clientes não cadastrados.

Tabela 14: Regra de negócio – Clientes não cadastrados.

Clientes não cadastrados (RN12)	
Descrição	O cliente não cadastrado só poderá efetuar compras a vista em dinheiro.

Fonte: Guilherme Post

Regra de negócio ao atendente não cadastrado.

Tabela 15: Regra de negócio – Atendente não cadastrado.

Atendente não Cadastrado (RN13)	
Descrição	Para emitir pedido de venda e realizar consultas o atendente terá que ser cadastrado.

Fonte: Guilherme Post

Regra de negócio para o recebimento de notas fiscais em desacordo.

Tabela 16: Regra de negócio – Recebimento de nota fiscal em desacordo.

Recebimento de Notas Fiscais Desacordo (RN14)	
Descrição	Após recebimento da Nota Fiscal de entrada caso seja identificada a falta de itens conforme solicitado no pedido de compra, ou os itens faltantes não constarem na NF o usuário poderá alterar a quantidade recebida.

Fonte: Guilherme Post

Regra de negócio ao cadastrar produto.

Tabela 17: Regra de negócio – Cadastro de produtos.

Cadastro de Produtos (RN15)	
Descrição	Para cadastrar o produto deverá: <ul style="list-style-type: none"> ✓ Informar a descrição do produto; ✓ Selecionar a categoria a qual pertencerá o produto; ✓ Marcar a unidade de medida; ✓ Deverá informar o preço de custo do produto, caso seja novo; ✓ Informar a margem de lucro; ✓ Deverá informar a quantidade mínima que deve ter em estoque.

Fonte: Guilherme Post

Regra de negócio ao cadastrar categoria.

Tabela 18: Regra de negócio – Cadastro de categoria.

Cadastro de Categoria (RN16)	
Descrição	Deverá informar: <ul style="list-style-type: none"> ✓ A descrição da categoria; ✓ Confirmar cadastro.

Fonte: Guilherme Post

Regra de negócio ao cadastrar pagamento.

Tabela 19: Regra de negócio – Cadastrar de pagamentos.

Cadastro de Pagamento (RN17)

Descrição	<p>Deverá informar:</p> <ul style="list-style-type: none"> ✓ Deverá informar a ordem de compra; ✓ Deverá confirmar os dados: comprador, forma de pagamento, condição de pagamento, fornecedor, data de emissão e valor total com impostos; ✓ Informará o numero da NF de entrada; ✓ Confirmar cadastro.
-----------	---

Fonte: Guilherme Post

Regra de negócio ao cadastrar forma de pagamento.

Tabela 20: Regra de negócio – Cadastro de forma de pagamento.

Cadastro de Forma de Pagamento (RN18)

Descrição	<p>Deverá informar:</p> <ul style="list-style-type: none"> ✓ A descrição da forma de pagamento; ✓ Confirmar cadastro.
-----------	---

Fonte: Guilherme Post

Regra de negócio ao cadastrar condição de pagamento.

Tabela 21: Regra de negócio – Cadastrar condição de pagamento.

Cadastro de Condição de Pagamento (RN19)

Descrição	<p>Deverá informar:</p> <ul style="list-style-type: none"> ✓ A descrição da condição de pagamento; ✓ Confirmar cadastro.
-----------	--

Fonte: Guilherme Post

Regra de negócio ao emitir venda com cartão.

Tabela 22: Regra de negócio – Cadastro de vendas com cartão.

Cadastro de vendas com cartão (RN20)

Descrição	<p>Deverá informar:</p> <ul style="list-style-type: none"> ✓ O cartão a ser utilizado para pagamento; ✓ Número do cartão; ✓ Nome do titular do cartão; ✓ Validade do cartão; ✓ Código de autorização; ✓ Numero de parcelas; ✓ Valor total da compra; ✓ Selecionar o caixa.
-----------	--

Fonte: Guilherme Post

Regra de negócio ao solicitar relatório de contas a pagar.

Tabela 23: Regra de negócio – Gerar relatórios contas a pagar.

Gerar relatórios contas a pagar (RN23)

Descrição	<p>Deverá informar no mínimo:</p> <ul style="list-style-type: none"> ✓ O período da pesquisa.
-----------	--

Fonte: Guilherme Post

Regra de negócio ao emitir vendas com cheques.

Tabela 24: Regra de negócio – Cadastro de vendas com cheque.

Cadastro de vendas com cheque (RN21)

Descrição	Deverá informar:
	<ul style="list-style-type: none"> ✓ Informa a quantidade de folhas a serem digitadas; ✓ Valor total dos cheques; ✓ Selecionar o cliente; ✓ Nome do emissor do cheque; ✓ Data de entrada; ✓ Valor da total do cheque; ✓ Nome do Banco; ✓ Agência; ✓ Numero do cheque; ✓ Data para deposito.

Fonte: Guilherme Post

Regra de negócio ao cadastrar senha.

Tabela 25: Regra de negócio – Cadastro de senhas.

Cadastro de Senhas (RN22)

Descrição	Deverá informar:
	<ul style="list-style-type: none"> ✓ O funcionário; ✓ Senha, letras ou números, no máximo 6 caracteres; ✓ Os módulos que poderão ter acessos; ✓ O que poderá fazer excluir, alterar, cadastrar e consultar; ✓ Quais janelas terão acessos.

Fonte: Guilherme Post

Regra de negócio ao solicitar relatórios.

Tabela 26: Regra de negócio – Gerar relatório de produtos em falta no estoque.

Gerar relatórios produtos em falta no estoque (RN23)

Descrição	Deverá informar no mínimo:
	<ul style="list-style-type: none"> ✓ O período da pesquisa.

Fonte: Guilherme Post

Regra de negócio ao solicitar relatórios de contas a pagar.

Tabela 27: Regra de negócio – Gerar relatórios contas a receber.

Gerar relatórios contas a receber (RN23)

Descrição	Deverá informar no mínimo:
	<ul style="list-style-type: none"> ✓ O período da pesquisa.

Fonte: Guilherme Post

Regra de negócio ao solicitar relatório de fluxo de caixa.

Tabela 28: Regra de negócio – Gerar relatórios fluxo de caixa.

Gerar relatórios fluxo de caixa (RN23)

Descrição	Deverá informar no mínimo:
	<ul style="list-style-type: none"> ✓ O período da pesquisa.

Fonte: Guilherme Post

3.4. Levantamento de Requisitos

O levantamento de requisitos teve inicio após analise do negocio e o entendimento dos problemas enfrentados e apresentados pelo cliente.

Através de reuniões realizadas com analista entre proprietário, o Sr. Emmer Post e demais envolvidos, foram apresentadas as necessidades e solicitações do que o sistema deve oferecer para atender as necessidades dos usuários.

Com as necessidade e solicitações apresentadas foi possível destacar as funções mais importantes que o sistema precisará disponibilizar que são apresentadas na tabela 29.

Requisitos funcionais:

Tabela 29: Requisitos do Sistema

Requisito	Realizador
Cadastrar Clientes	Atendente
Cadastrar Funcionário	Gerente
Cadastrar Produto	Atendente
Aprovação de Pedido de Compra	Gerente
Envio de aviso	Sistema
Cadastrar cheques	Atendente
Cadastrar nota fiscal de entrada	Atendente
Enviar aviso	Sistema
Cadastrar contas a pagar	Atendente
Cadastrar contas a receber	Sistema
Enviar aviso	Sistema
Cadastrar nota fiscal de saída	Sistema
Enviar aviso	Sistema
Calcular fatura	Sistema
Gerar faturas	Sistema
Imprimir fatura	Sistema
Baixar fatura	Atendente
Imprimir Relatório de movimentações do período	Atendente/Gerente

Fonte: Guilherme Post

Após identificar as funcionalidades do sistema o usuário terá acesso conforme sua permissão criada pelo usuário do RH.

Os requisitos não funcionais foram definidos para tornar o sistema mais ágil e seguro, garantindo a satisfação do cliente. Os requisitos não funcionais são apresentados na tabela 30.

Tabela 30: Requisitos não Funcionais

Requisitos Não Funcionais
O tempo de resposta do sistema não deve ultrapassar 60 segundos
A base de dados deve ser protegida para acesso apenas de usuários autorizados através de <i>login</i> e senha
O tempo de desenvolvimento não deve ultrapassar 3 meses
Deve ser desenvolvido para ter facilidade de uso e manutenibilidade

Fonte: Guilherme Post

3.4.1. Modelo de Domínio do Sistema

Com o modelo de domínio fica mais fácil a visualização e o entendimento do sistema a ser desenvolvido para empresa tratando os objetos específicos do negócio como classes do sistema, com isso é possível identificar os problemas a serem solucionados. No modelo de domínio é apresentado uma visão geral de todo o sistema, mostrando os atributos, métodos que o sistema deverá executar.

O modelo de domínio apresentado neste projeto apresenta as tabelas e seus relacionamentos, dentro de cada tabela é possível identificar o nome da tabela, os atributos que possui. A tabela cliente possui vários atributos como, *clie_nome*, *clie_cpf*, esses campos serão responsáveis por permitir que o sistema armazene as informações como nome do cliente, CPF, data de nascimento e outros. Os relacionamentos foram ajustados conforme a regra de negocio estabelecida com os usuários. Exemplo: um cliente pode ter 1 (uma) ou várias (*) vendas, mas 1 (uma) venda é de apenas 1 (um) cliente. O modelo de domínio apresentado neste projeto é visto na figura 17.

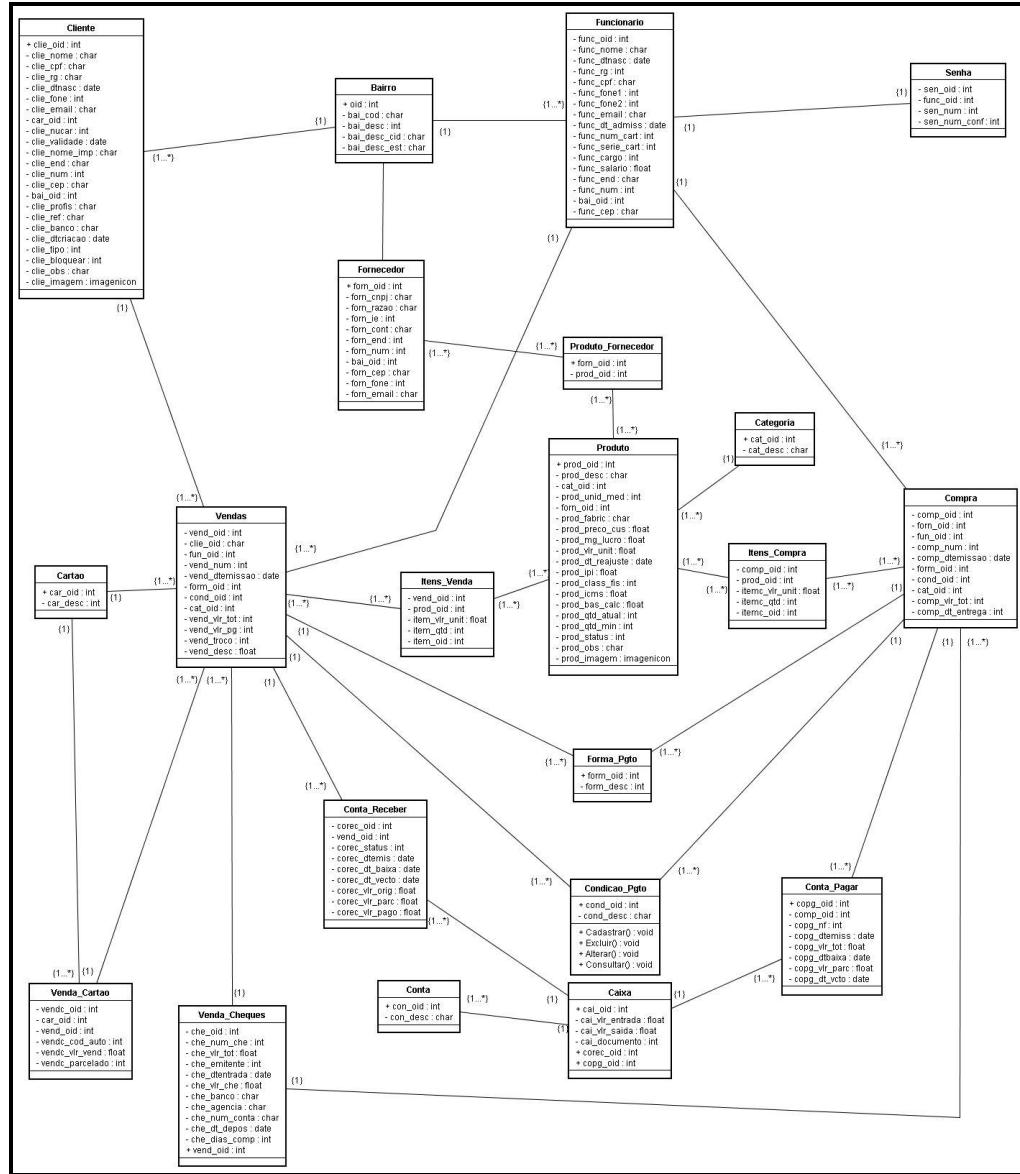


Figura 18: Modelo de domínio

Fonte: Guilherme Post.

3.4.2. Encontrar atores

Nesta etapa são definidos os atores do sistema. Os atores são todos os indivíduos que interagem com o sistema.

Um ator receberá o nome de “atendente” que será responsável por informar e manter os cadastros do sistema, o outro receberá o nome de “gerente”, responsável pela análise de relatórios e tomada de decisões durante as atividades diárias. A tabela 31 apresenta a definição dos atores do sistema.

Tabela 31: Definição dos atores do sistema.

Atores		
Nº de Ordem	Nome do Ator	Definição
1	Atendente	É responsável por manter o cadastro de clientes, produtos, vendas, categoria, lançamento de vendas com cheque, cartão, e emitir relatórios.
2	Gerente	Responsável por analisar os pagamentos, compras, vendas, estoque, através de relatórios.

Fonte: Guilherme Post.

3.4.3. Encontrar caso de uso

Após analisar o modelo de domínio e com as entrevistas realizadas com o cliente é possível identificar os casos de uso do sistema apresentados logo abaixo. Casos de usos é a forma a qual os atores utilizaram para interagir com o sistema.

Na tabela 32 são apresentados os casos de uso do sistema.

Tabela 32: Definição dos casos de uso.

Casos de Uso		
Nº de Ordem	Nome do Caso de Uso	Definição
1	Manter cadastro de cliente	Cadastrar os dados dos clientes que serão armazenados no banco de dados do sistema.
2	Manter cadastro de funcionário	Cadastrar os dados dos funcionários que serão armazenados no banco de dados do sistema.
3	Manter cadastro de fornecedor	Cadastrar os dados dos fornecedores que serão armazenados no banco de dados do sistema.
4	Manter cadastro de produtos	Operação realizada para manter os dados do produto no sistema.
5	Manter cadastro de vendas	É a operação onde ocorre a emissão do pedido de venda, serão armazenados os dados das vendas no banco de dados do sistema, que irá gerar as contas a receber. Com este caso de uso é necessário informar dados obrigatórios para manter os dados das vendas.
6	Manter cadastro de compras	É a operação onde ocorre a emissão do pedido de compra realizada pela empresa.
7	Manter cadastro de contas a pagar	É responsável por manter o cadastramento dos pagamentos, que deverão ser feitos aos fornecedores e banco, é informado os dados que serão armazenados no banco de dados.

Nº de Ordem	Nome do Caso de Uso	Definição
8	Manter cadastro de nota fiscal de entrada	Operação responsável por manter atualizado o estoque, mantendo as informações de quantidade de produto que está entrando, preço unitário dos produtos que consta na nota fiscal, fornecedores que estão fazendo entrega, no prazo estabelecido e demais valores da nota.
9	Manter cadastro de categoria	Cadastrar a categoria a qual pertencerá vários produtos.
10	Manter cadastro de vendas com cheque	Operação responsável por controlar as vendas realizadas com cheques, onde serão informados os dados do cliente, conta bancaria, valores, e prazos.
11	Manter cadastro de vendas com cartão	Operação responsável por controlar as vendas realizadas com cartão, onde serão informados os dados do cliente, conta bancaria, numero do cartão, validade, valores, parcelas e prazos.
12	Manter cadastro de bairros	Responsável por manter o cadastro de bairros.
13	Manter cadastro de forma de pagamento	Operação que mantém o cadastro da forma de pagamento que poderá ser utilizada no pedido de compra ou de venda.
14	Manter cadastro condição de pagamento	Operação que mantém o cadastro da condição de pagamento que poderá ser utilizada no pedido de compra ou de venda.
15	Manter cadastro de conta	Responsável por cadastrar a conta a ser lançado o pagamento de contas.
16	Manter cadastro de senhas	Operação que mantém o cadastro de senhas de acessos, para os usuários interagirem com o sistema.
17	Consultar compras	Operação na qual o usuário atendente ou gerente visualiza as compras realizadas para determinados fornecedores.
18	Consultar clientes	Responsável por apresentar os dados dos clientes, podendo ser filtrada a pesquisa por nome, CPF ou bairro.
19	Consultar fluxo de caixa	Operação em que o usuário gerente acompanha o movimento do fluxo de caixa da empresa.
20	Consultar vendas	Apresenta os dados das vendas feitas aos clientes.
21	Consultar produtos	Apresenta a lista de produtos em estoque, mostrando a quantidade, preço unitário, fornecedor e quantidade mínima.
22	Manter estoque	Operação que possibilita selecionar determinados produtos através de fornecedores, categoria ou produto para realizar os reajustes de preços e atualizar as informações e visualizar a situação dos produtos no estoque.
23	Consultar pagamento	Possibilita informar parâmetros para filtrar a pesquisa dos pagamentos.
24	Consultar recebimento	Operação que possibilita informar parâmetros para filtrar a pesquisa das contas a receber.

Nº de Ordem	Nome do Caso de Uso	Definição
25	Controle de vendas com cartão	Responsável por listar os dados das vendas feitas com cartão. Podendo ser apresentada vendas por determinada bandeira de cartão.
26	Manter cadastro de cartão	Operação que mantém o cadastro dos cartões oferecidos pelas financeiras.
27	Controle de vendas feitas com cheques	Responsável por listar dos dados das vendas feitas com cheque. Podendo filtrar as vendas feitas por cliente, data de depósito, número do cheque.
28	Gerar relatório de contas a receber	Operação responsável por filtrar as contas a receber geradas pelas vendas efetuadas aos clientes. Podendo visualizar os dias em atraso, e as contas já quitadas.
29	Gerar relatório de contas a pagar	Operação responsável por apresentar as contas a serem pagas ou já quitadas pela empresa, mostrando ainda dias em atraso caso haja.
30	Gerar relatório de produtos em falta no estoque	Responsável por apresentar os itens em falta no estoque, podendo ser filtrado por fornecedor.
31	Gerar relatório de fluxo de caixa	Apresenta o fluxo de contas a receber e a pagar em um determinado período.
32	Efetuar login	Responsável por validar o acesso ao sistema.

Fonte: Guilherme Post.

3.4.4. Prioridade dos Casos de Uso

Para desenvolver o sistema foi necessário identificar a prioridade da implementação dos cada caso de uso. Foram selecionados os casos de uso mais importantes para enfatizar a e priorizar os casos de uso durante o desenvolvimento do projeto. Através de uma análise concluiu-se que a prioridade mais adequada seria a detalhada na tabela 33.

Tabela 33: Priorização de Casos de Uso

Módulo	Prioridade	Caso de Uso
Comercial	1	Manter cadastro de endereço
		Manter cadastro de cliente
		Manter cadastro de venda
Suprimentos	2	Manter cadastrar nota fiscal de entrada
		Manter cadastrar de produto
		Manter cadastro de compra

Módulo	Prioridade	Caso de Uso
Financeiro	3	Manter cadastro de conta
		Manter cadastro de forma de pagamento
		Manter cadastro de condição de pagamento
		Manter cadastro de contas a pagar
		Manter controle de vendas com cheque
		Manter controle de vendas com cartão
Rh	4	Manter cadastro de funcionário
		Manter cadastrar de senha

Fonte: Guilherme Post.

3.4.5. Definir Caso de Uso

Nesta etapa são apresentados na figura 18 todos os casos de uso do sistema e a interação com os atores assim como cada funcionalidade acordada durante o levantamento de requisitos. Todos os casos de uso são ligados ao ator que possui interação, e para utilizar cada caso de uso é necessário estar logado no sistema, devido a esta situação os casos de uso possuem uma ligação com um caso de uso em comum que é o de efetuar *login*.

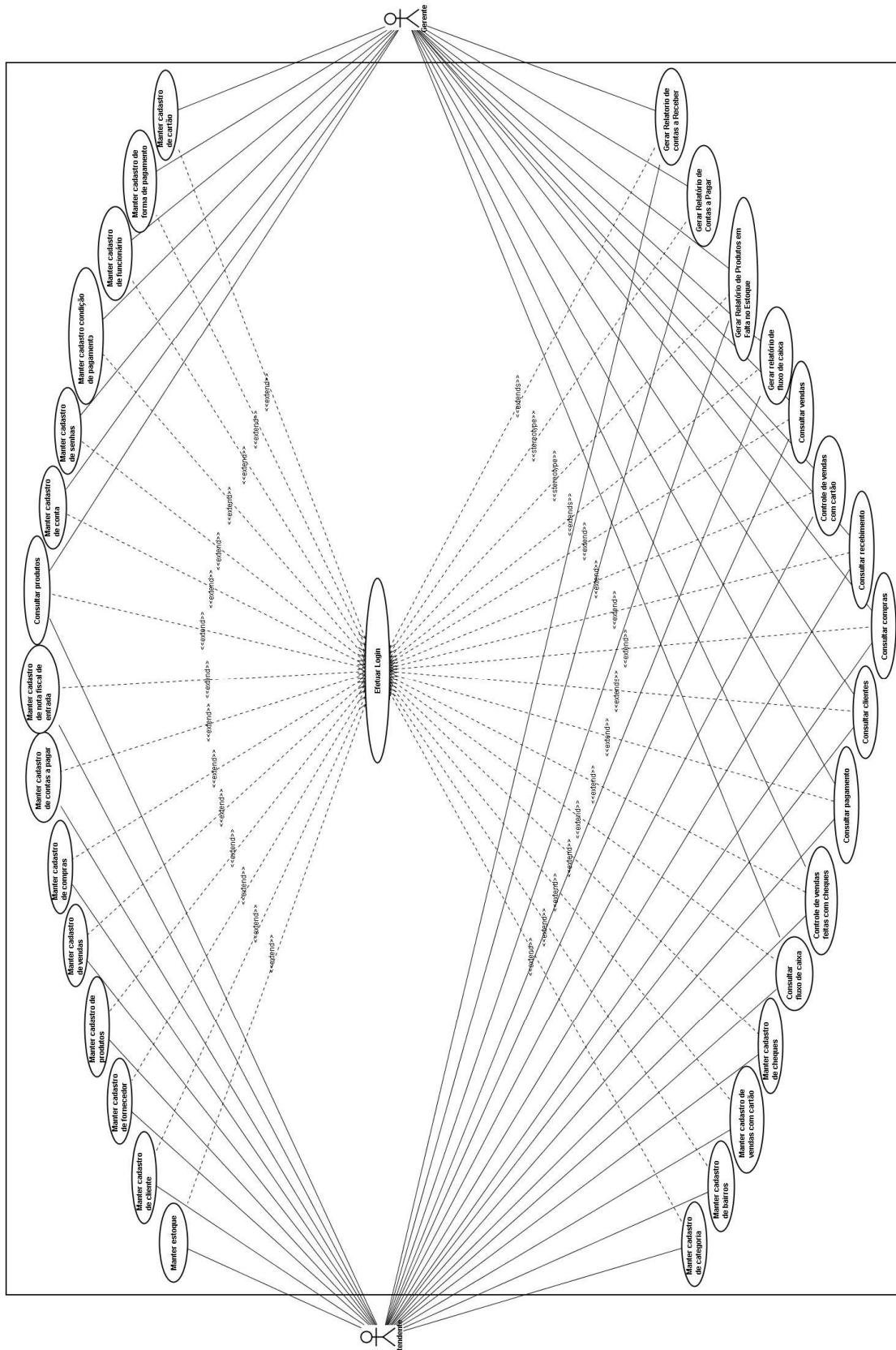


Figura 19: Diagrama de Casos de Uso de Contexto
Fonte: Guilherme Post.

3.4.6. Detalhar Caso de Uso

Neste momento é apresentado o detalhamento de todos os casos de uso, proporcionando assim o entendimento do sistema e de cada funcionalidade, assim como a interação de cada ator com o sistema. Nas tabelas a seguir o detalhamento dos casos de uso.

Caso de uso efetuar *login*.

Tabela 34: Caso de Uso – Efetuar *login*

Nome:	Efetuar login
Objetivo:	Permitir usuários acessarem os módulos do sistema, comercial, suprimentos, financeiro, recursos humanos.
Ator Primário:	Atendente
Ator Secundário:	Gerente
Pré-Condição:	Funcionário tenha senha e login cadastrado.
Pós-Condição:	Os dados da senha do funcionário cadastrados no banco de dados.
Ativação:	Ao clicar no botão “Acessar” na tela de cadastro de senhas.
Fluxo principal:	1. Usuário seleciona o usuário, informa a senha; 2. Sistema consulta senha e login; 7.1 Senha e login valida, sistema apresenta menu principal; 7.2 Senha invalida, sistema apresenta mensagem, “Senha ou login invalida! consulte administrador.”.

Fonte: Guilherme Post

Caso de uso manter cadastro de funcionários.

Tabela 35: Caso de Uso – Manter cadastro de funcionário

Nome:	Manter cadastro de funcionário
Objetivo:	Serão cadastrados os dados dos funcionários.
Ator Primário:	Gerente
Ator Secundário:	Não tem
Pré-Condição:	Ter em mãos os dados do funcionário e o atendente estar logado no sistema.
Pós-Condição:	Os dados do funcionário cadastrados no banco de dados.
Ativação:	Após clicar no botão “Salvar” na tela de cadastro de funcionários.
Fluxo principal:	1. Usuário seleciona na tela principal o menu Recursos Humanos em cadastrar funcionário; 2. Sistema apresenta tela de cadastro de funcionários com os campos desabilitados; 3. Usuário clica no botão “Novo funcionário”; 4. Sistema habilita os campos da tela; 5. Usuário precisará preencher todos os campos obrigatórios informando os dados solicitados para cadastro; 6. Usuário pressionará o botão “Salvar”; 7. Sistema analisará dados informados pelo Usuário; 7.1 Se o cadastro tiver informações erradas aparecerá uma mensagem informando onde esta errada e setará o foco no campo “CPF invalido” - foco no campo do CPF; 7.2 Caso esteja correto o cadastro aparecerá uma mensagem informando “Gravação realizada com sucesso”; 8. Usuário clica na Mensagem no botão OK! 9. Sistema finaliza processo gravando os dados no banco de dados; 10. Sistema limpa os campos; 11. Sistema desabilita os campos.

Fonte: Guilherme Post

Caso de uso manter cadastro de senhas.

Tabela 36: Caso de Uso – Manter cadastro de senhas

Nome:	Manter cadastro de senhas
Objetivo:	Usuário responsável por cadastrar senha para cada funcionário determinando os acessos e restrições.
Autor Primário:	Gerente
Autor Secundário:	Não possui
Pré-Condição:	Ter em mãos os dados do funcionário e o atendente estar logado no sistema.
Pós-Condição:	Os dados da senha do funcionário cadastrados no banco de dados.
Ativação:	Ao clicar no botão “Salvar” na tela de cadastro de senhas.
Fluxo principal:	<p>3. Usuário seleciona na tela principal o menu Financeiro a opção cadastrar senha;</p> <p>4. Sistema apresenta tela de cadastro de senhas com os campos desabilitados;</p> <p>5. Usuário clica no botão “Nova Senha”;</p> <p>6. Sistema habilita campos;</p> <p>7. Usuário seleciona o funcionário;</p> <p>8. Usuário informa a senha com seis caracteres;</p> <p>9. Sistema valida senha;</p> <p> 7.3 Senha valida, sistema seta foco no próximo campo;</p> <p> 7.4 Senha invalida, sistema apresenta mensagem, “Senha não cadastrada, informe nova senha!”.</p> <p>10. Usuário define o que poderá ser acessado pelo novo usuário:</p> <p> 8.1 Marca os módulos que o usuário terá acesso;</p> <p> 8.2 Marca as opções que os usuários poderão fazer (alterar, excluir, cadastrar e consultar);</p> <p> 8.3 Marca quais janelas poderá acessar, nos módulos financeiros, suprimentos, comercial, recursos humanos.</p> <p>11. Usuário clica na mensagem “botão Salvar”;</p> <p>12. Sistema finaliza processo gravando os dados no banco de dados;</p> <p>13. Sistema apresenta mensagem “Gravação realizada com sucesso”;</p> <p>14. Usuário clica na Mensagem no botão OK!</p> <p>15. Sistema limpa os campos;</p> <p>16. Sistema desabilita os campos.</p>

Fonte: Guilherme Post

Caso de uso consultar produtos.

Tabela 37: Caso de Uso – Consultar produtos

Nome:	Consultar produtos
Objetivo:	Apresentar os dados dos produtos cadastrados no banco de dados.
Autor Primário:	Atendente.
Autor Secundário:	Gerente.
Pré-Condição:	O atendente/gerente logado no sistema.
Pós-Condição:	Apresentar as informações dos produtos ao atendente/gerente.
Ativação:	Após usuário seleciona a categoria do produto.
Fluxo principal:	<p>1. Usuário seleciona na tela principal o menu Suprimentos a opção Estoque, em seguida Consultar de Produtos;</p> <p>2. Sistema apresenta a tela de consulta de produtos;</p> <p>3. Usuário seleciona uma categoria de produtos;</p> <p>4. Sistema consulta categoria;</p> <p> 4.1 Se não houver produtos cadastrados para a categoria selecionada, sistema apresenta mensagem “Não há produtos cadastrados para esta categoria!”;</p> <p> 4.2 Se houver produtos, sistema apresenta os dados dos produtos;</p>

Fonte: Guilherme Post

Caso de uso cadastro de compras.

Tabela 38: Caso de Uso – Manter cadastro de compras

Nome:	Manter cadastro de compra.
Objetivo:	Manter o cadastro de compras realizadas pela empresa, controlando as compras de produtos através dos fornecedores.
Autor Primário:	Atendente
Autor Secundário:	Não possui.
Pré-Condição:	O atendente deve receber a ordem do gerente para efetuar o cadastro.
Pós-Condição:	O pedido de compra estará liberado ao fornecedor.
Ativação:	O usuário selecionará a opção “salvar” tela de cadastro de compras.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Suprimentos opção Compra, em seguida Emitir Compra; 2. Sistema apresenta tela de emissão de compras com os campos desabilitados; 3. Usuário clica no botão “Nova Compra”; 4. Sistema habilita campos; 5. Usuário precisará preencher os campo CNPJ ou razão social e pressionar a tecla “ENTER”; 6. Sistema consulta CNPJ ou Razão Social; <ul style="list-style-type: none"> 6.1. Se CNPJ cadastrado apresenta a Razão Social; 6.2. Se Razão Social cadastrada apresenta CNPJ; 6.3. Se CNPJ ou Razão Social não cadastrado sistema apresenta mensagem “Fornecedor não cadastrado!”; 7. Usuário informará funcionário, forma de pagamento, condição de pagamento; 8. O usuário ainda precisará selecionar os itens da compra, poderá através da opção categoria selecionar somente os itens da categoria selecionada; 9. Usuário ao selecionar o item desejado irá pressionar o botão “Inserir”, precisará informar a quantidade e preço unitário a ser pago; 10. Se o usuário decidir retirar o produto selecionado, deverá selecionar o item e pressionar o botão “Remover”; 11. Usuário finaliza a compra clicando no botão “Salvar”; 12. Sistema analisará dados informados pelo Usuário; <ul style="list-style-type: none"> 12.1. Se não houver condição de pagamento sistema apresenta mensagem “Informe a condição de pagamento!”; 12.2. Se não houver forma de pagamento sistema apresenta mensagem “Informe a forma de pagamento!”; 12.3. Se não houver itens, sistema apresenta mensagem “Informe ao menos um item, com embalagem mínima e preço unitário!”; 12.4. Caso esteja correto o cadastro aparecerá uma mensagem informando “Gravação realizada com sucesso”; 13. Usuário clica na Mensagem no botão “OK”!; 14. Sistema finaliza processo gravando os dados no banco de dados; 15. Sistema limpa os campos; 16. Sistema desabilita os campos.

Fonte: Guilherme Post

Caso de uso manter pedido de compra.

Tabela 39: Caso de Uso – Consultar pedido de compra

Nome:	Consultar pedido de compra
Objetivo:	Permitirá o usuário atendente ter acesso aos dados dos pedidos de compras realizados pela empresa aos fornecedores.
Autor Primário:	Atendente.
Autor Secundário:	Não possui.
Pré-Condição:	O atendente/gerente deve estar logado no sistema.
Pós-Condição:	Apresentar as informações das compras feitas aos fornecedores para o atendente/gerente.
Ativação:	Informar o CNPJ e pressionar a tecla “ENTER”, selecionar o fornecedor pela razão social, ou selecionar umas das opções “Pedidos não entregue” ou “Pedidos já entregue”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Suprimentos a opção Compra, em seguida Consultar Compras; 2. Sistema apresenta a tela de consulta de compras com os campos desabilitados; 3. Para consultar as compras o usuário poderá informar o CNPJ e pressionar a tecla “ENTER”, selecionar o fornecedor pela razão social, ou selecionar as opções “Pedidos já entregue” ou “Pedidos não entregue”; <ol style="list-style-type: none"> 3.1 Usuário informa o CNPJ e pressiona a tecla “ENTER”; 3.2 Sistema consulta CNPJ; <ol style="list-style-type: none"> 3.2.1 CNPJ valido, sistema apresenta dados das compras; 3.2.2 CNPJ inválido, sistema apresenta mensagem “CNPJ inválido!”; 3.3 Usuário seleciona o fornecedor pela razão social; 3.4 Sistema consultar fornecedor; <ol style="list-style-type: none"> 3.4.1 Fornecedor sem pedidos de compra cadastrados apresenta mensagem “Não há pedidos para este fornecedor!”; 3.4.2 Fornecedor com pedidos cadastrados, sistema apresenta os dados das compras para o fornecedor selecionado; 3.5 Usuário seleciona a opção “Pedidos já entregue”; 3.6 Sistema consulta todos os pedidos no banco de dados que possuem a data de entrega; <ol style="list-style-type: none"> 3.6.1 Se não houver pedidos com data de entrega sistema apresenta mensagem “Não há pedidos entregue”. 3.6.2 Se houver pedidos com data de entrega, sistema apresenta os dados das compras;

Fonte: Guilherme Post

Caso de uso manter cadastro de nota fiscal de entrada.

Tabela 40: Caso de Uso – Manter Cadastro de Nota Fiscal de Entrada

<u>Nome:</u>	Manter cadastro de nota fiscal de entrada
<u>Objetivo:</u>	Cadastrar as notas fiscais de entrada.
<u>Autor Primário:</u>	Atendente.
<u>Autor Secundário:</u>	Não possui.
<u>Pré-Condição:</u>	O atendente estar logado no sistema e ter em mãos os dados da nota fiscal de entrada.
<u>Pós-Condição:</u>	Os dados da nota fiscal de entrada estarão cadastrados no sistema.
<u>Ativação:</u>	Ao pressionar o botão “Salvar” na tela de cadastro de compras.
<u>Fluxo principal:</u>	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Suprimentos opção Cadastrar Compra; 2. Usuário pressiona no botão “Nova Compra”; 3. Sistema habilita campos; 4. Usuário informa o numero do pedido de compra que consta na nota fiscal do fornecedor; 5. Sistema consulta número do pedido de compra; <ul style="list-style-type: none"> 5.1 Número válido, sistema apresenta os dados da compra; 5.2 Número inválido, sistema apresenta mensagem “Número de pedido de compra não cadastrado!”; 6. Usuário poderá alterar as informações do pedido de compra caso necessário; 7. Usuário informa a data do recebimento da nota fiscal; 8. Usuário pressiona o botão “Salvar”; 9. Sistema analisa os dados informados; <ul style="list-style-type: none"> 9.1 Data inválida do recebimento, sistema apresenta mensagem “Data inválida!”; 9.2 Sem data de recebimento, sistema apresenta mensagem “Informe a data de recebimento da nota fiscal!”; 9.3 Quantidade não informada, apresenta mensagem “Informe a quantidade!”; 9.4 Preço não informado, apresenta mensagem “Informe preço unitário!”; 9.4 Se dados validos, sistema apresenta mensagem “Gravação realizada com Sucesso!”; 10. Usuario clica na mensagem no botão “OK!”; 11. Sistema finaliza processo gravando os dados no banco de dados; 12. Sistema desabilita os campos.

Fonte: Guilherme Post

Caso de uso manter cadastro de clientes.

Tabela 41: Caso de Uso – Manter cadastro de clientes

<u>Nome:</u>	Manter cadastro de clientes
<u>Objetivo:</u>	Manter o cadastro de cliente.
<u>Ator Primário:</u>	Atendente
<u>Ator Secundário:</u>	Não possui
<u>Pré-Condição:</u>	O atendente deve estar logado no sistema e de ter em mãos os dados do cliente.
<u>Pós-Condição:</u>	Os dados do cliente estarão cadastrados no sistema.
<u>Ativação:</u>	Clicar no botão “Salvar” na tela de cadastro de clientes.
<u>Fluxo principal:</u>	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Comercial a opção Cliente, em seguida Cadastrar Cliente; 2. Sistema apresenta a tela de cadastro de clientes com os campos desabilitados; 3. Usuário pressiona o botão “Novo cliente”; 4. Sistema habilita campos; 5. Usuário informa os dados do cliente e pressiona no botão “Salvar”; 6. Sistema verifica o CPF/CNPJ/Nome do cliente já esta cadastrada na base de dados; <ul style="list-style-type: none"> 6.1 Se CPF/CNPJ/Nome já existe sistema apresenta mensagem “Cliente já cadastrado!”; 6.2 Se CPF/CNPJ/Nome não existe sistema armazena os dados no sistema e apresenta mensagem “Gravação realizada com Sucesso!”; 7. Usuário clica no botão “Ok” da mensagem; 8. Sistema desabilita campos.

Fonte: Guilherme Post

Caso de uso manter cadastro de fornecedores.

Tabela 42: Caso de Uso – Manter cadastro de fornecedores

<u>Nome:</u>	Manter cadastro de fornecedores
<u>Objetivo:</u>	Manter o cadastro dos dados dos fornecedores.
<u>Ator Primário:</u>	Atendente
<u>Ator Secundário:</u>	Não possui.
<u>Pré-Condição:</u>	Ter em mãos os dados do fornecedor e o atendente estar logado no sistema.
<u>Pós-Condição:</u>	Os dados do fornecedor cadastrados no banco de dados.
<u>Ativação:</u>	Após clicar botão “salvar” na tela de cadastro de fornecedores.
<u>Fluxo principal:</u>	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Suprimentos a opção Fornecedor, cadastrar fornecedor; 2. Sistema apresenta a tela de cadastro de fornecedores com os campos desabilitados; 3. Usuário clica no botão “Novo Fornecedor”; 4. Sistema habilita campos; 5. Usuário precisará preencher todos os campos obrigatórios informando os dados solicitados para cadastro; 6. Usuário pressionará o botão “Salvar”; 7. Sistema analisará dados informados pelo Usuário; <ul style="list-style-type: none"> 7.1 Se o cadastro constar informações erradas aparecerá uma mensagem informando onde esta errada. Exemplo: “CNPJ invalida” o foco será setado no campo errado; 7.2 Caso esteja correto o cadastro aparecerá uma mensagem informando “Gravação realizada com sucesso”; 8.Usuário clica na Mensagem no botão OK! 9. Sistema finaliza processo gravando os dados no banco de dados; 10. Sistema limpa os campos; 11.Sistema desabilita os campos.

Fonte: Guilherme Post

Caso de uso manter pedido de venda.

Tabela 43: Caso de Uso – Manter pedido de venda

<u>Nome:</u>	Manter pedido de venda
<u>Objetivo:</u>	Efetuar o cadastro de vendas solicitadas pelo cliente.
<u>Autor Primário:</u>	Atendente.
<u>Autor Secundário:</u>	Não possui
<u>Pré-Condição:</u>	O cliente deve ao menos solicitar um item para comprar e atendente logado no sistema.
<u>Pós-Condição:</u>	Os dados da venda estarão cadastrados no sistema.
<u>Ativação:</u>	Ao clicar no botão “Salvar” na tela de emissão de vendas.
<u>Fluxo principal:</u>	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Comercial opção Venda, em seguida Emittir Venda; 2. Sistema apresenta a tela de emissão de vendas; 3. Usuário pressiona o botão “Novo”; 4. Sistema habilita campos; 5. Usuário precisará preencher os campo CPF ou nome completo e pressionar a tecla “ENTER”; 4. Sistema analisará dados informados pelo Usuário; <ol style="list-style-type: none"> 4.1 Se cliente não cadastrado: <ol style="list-style-type: none"> 4.1.1 Sistema emite mensagem “Cliente não cadastrado!”; 4.1.2 Sistema apresenta mensagem “Deseja cadastrar cliente?” <ol style="list-style-type: none"> 4.1.2.1 Sim, sistema apresenta tela de cadastro de cliente; 4.1.2.2 Não, sistema seta foco no campo CPF; 4.2 Cliente Cadastrado: <ol style="list-style-type: none"> 4.2.1 Sistema apresenta dados do cliente; 5. Usuário informa a forma de pagamento; 6. Usuário informa a condição de pagamento; 7. Usuário seleciona o produto e pressiona o botão “Inserir”; 8. Sistema inserir produto; 9. Usuário informa a quantidade que o cliente solicitou e pressiona a tecla “ENTER”; 10. Sistema calcular o total e apresenta valor total da compra e do item; 11. Usuário clica no botão “Salvar” na tela de emissão de vendas; 12. Sistema grava os dados da venda no banco de dados e apresenta mensagem “Gravação realizada com sucesso!”. 13. Usuário pressiona no botão OK da mensagem; 14. Sistema desabilita campos.

Fonte: Guilherme Post

Caso de uso consultar pedido de venda.

Tabela 44: Caso de Uso – Consultar venda

<u>Nome:</u>	Consultar pedido de venda
<u>Objetivo:</u>	Consultar os dados das vendas solicitadas pelos clientes.
<u>Autor Primário:</u>	Atendente
<u>Autor Secundário:</u>	Gerente
<u>Pré-Condição:</u>	Atendente/Gerente logado no sistema.
<u>Pós-Condição:</u>	O atendente/gerente terá as informações das vendas solicitadas pelos clientes.
<u>Ativação:</u>	Ao informar CPF e pressionar a tecla “ENTER”, ou clicar na opção “Todos os clientes”, ou pressionar o botão “Consultar”.
<u>Fluxo principal:</u>	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Comercial opção Vendas, em seguida Consultar Vendas; 2. Sistema apresenta tela de consulta aos dados das vendas; 3. Usuário poderá informar: <ol style="list-style-type: none"> 3.1 Usuário Informar CPF, ao pressionar a tecla “ENTER” sistema consulta CPF: <ol style="list-style-type: none"> 3.1.1 CPF inválido, sistema apresenta mensagem “CPF não cadastrado!”; 3.1.2 CPF válido, sistema apresenta os dados da venda emitidas para este CPF; 3.2 Usuário informa a data inicial e data final da pesquisa; 3.3 Usuário seleciona o poderá seleciona o cliente desejado ou deixar todos; 3.4 Sistema consulta todas as vendas emitidas nos períodos informado e apresenta dados; <ol style="list-style-type: none"> 3.4.1 Se não houver pedidos de venda emitidos sistema apresenta mensagem “Não há pedidos cadastrados no período informado!”; 3.5 Usuário poderá selecionar a opção todas as vendas; 3.6 Sistema apresenta os dados de todas as vendas armazenadas no banco de dados; 4. Usuário poderá selecionar um pedido na lista e visualizar na tabela logo abaixo os itens da venda.

Fonte: Guilherme Post

Caso de uso consultar clientes.

Tabela 45: Caso de Uso – Consultar clientes

Nome:	Consultar clientes
Objetivo:	Consultar os dados dos clientes armazenados no banco de dados através do cadastro de clientes.
Autor Primário:	Atendente.
Autor Secundário:	Gerente.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Apresentar as informações dos clientes ao atendente/gerente.
Ativação:	Ao informar o CPF ou nome do cliente e pressionar a tecla “ENTER”, ou ao selecionar o bairro ou opção “Todos os Clientes”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Comercial opção Cliente, em seguida Consultar Cliente; 2. Usuário informa o CPF e pressiona a tecla “ENTER”; 3. Sistema consulta CPF: <ul style="list-style-type: none"> 3.1 CPF válido, sistema apresenta dados dos clientes; 3.2 CPF inválido, sistema apresenta mensagem “CPF não cadastrado”; 4. Usuário informa o nome do cliente e pressiona a tecla “ENTER”; 5. Sistema consulta nome do cliente; <ul style="list-style-type: none"> 5.1 Cliente cadastrado, sistema apresenta os dados do cliente; 5.2 Cliente não cadastrado, sistema apresenta mensagem “Cliente não cadastrado!”; 6. Usuário seleciona o bairro; 7. Sistema consulta bairro; <ul style="list-style-type: none"> 7.1 Clientes não cadastrados, sistema apresenta mensagem “Não há clientes cadastrados para este bairro!”; 7.2 Clientes cadastrados, sistema apresenta os dados dos clientes conforme bairro selecionado; 8. Usuário poderá selecionar a opção “Todos os clientes”; 9. Sistema consulta no banco todos os clientes; <ul style="list-style-type: none"> 9.1 Clientes cadastrados, apresenta os dados de todos os clientes; 9.2 Cliente não cadastrados, apresenta mensagem “Não há clientes cadastrados!”.

Fonte: Guilherme Post

Caso de uso manter cadastro de condição de pagamento.

Tabela 46: Caso de Uso – Cadastrar condição de pagamento

Nome:	Manter cadastro de condição de pagamento.
Objetivo:	Manter o cadastro das condições de pagamentos que serão utilizadas nas emissão das vendas ou de compras.
Autor Primário:	Gerente.
Autor Secundário:	Não possui.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Os dados da condição de pagamento estarão cadastrados no sistema.
Ativação:	Ao pressionar no botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção Condição de Pagamento, em seguida Cadastrar Condição Pgto; 2. Sistema apresenta a tela de cadastro de condição de pagamento com os campos desabilitados; 3. Usuário pressiona o botão “Novo”; 4. Sistema habilita campos; 5. Usuário informa a descrição da condição de pagamento e pressiona no botão “Salvar”; 6. Sistema grava descrição no banco de dados, apresenta mensagem “Gravação realizada com Sucesso!”; 7. Usuário pressiona no botão “Ok”; 8. Sistema desabilita campos; 9. Usuário poderá pressionar os botões “Registro anterior” ou “Registro Posterior” para consultar as condições já cadastradas.

Fonte: Guilherme Post

Caso de uso manter cadastro de forma de pagamento.

Tabela 47: Caso de Uso – Manter cadastro de forma de pagamento

Nome:	Manter cadastro de forma de pagamento.
Objetivo:	Manter o cadastro das forma de pagamentos que serão utilizadas nas emissão das vendas ou de compras.
Autor Primário:	Gerente.
Autor Secundário:	Não possui.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Os dados da condição de pagamento estarão cadastrados no sistema.
Ativação:	Ao pressionar no botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção Forma de Pagamento, em seguida Cadastrar Forma de Pgto; 2. Sistema apresenta a tela de cadastro de forma de pagamento com os campos desabilitados; 3. Usuário pressiona o botão “Novo”; 4. Sistema habilita campos; 5. Usuário informa a descrição da forma de pagamento e pressiona no botão “Salvar”; 6. Sistema grava descrição no banco de dados, apresenta mensagem “Gravação realizada com Sucesso!”; 7. Usuário pressiona no botão “Ok”; 8. Sistema desabilita campos; 9. Usuário poderá pressionar os botões “Registro anterior” ou “Registro Posterior” para consultar as formas de pagamento já cadastradas.

Fonte: Guilherme Post

Caso de uso manter cadastro de produtos.

Tabela 48: Caso de Uso – Manter cadastro de produto

Nome:	Manter cadastro de produtos
Objetivo:	Cadastrar os produtos na empresa no sistema.
Autor Primário:	Atendente.
Autor Secundário:	Não possui.
Pré-Condição:	Atendente logado no sistema e dados do produto em mãos.
Pós-Condição:	O produto estará cadastrado no sistema.
Ativação:	Ao pressionar o botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Suprimentos opção Estoque, em seguida Cadastrar Produto; 2. Sistema apresenta a tela de cadastro de produtos com os campos desabilitados; 3. Usuário pressiona o botão “Novo”; 4. Sistema habilita campos; 5. Usuário informa os dados do produto; 6. Sistema consulta os dados dos produtos; <ul style="list-style-type: none"> 6.1 Dados incorretos, sistema seta o foco no campo onde está faltando ou está incorreto (Unidade de medida, quantidade mínima); 6.2 Sistema armazena os dados do produto; 7. Sistema apresenta mensagem “Gravação realizada com sucesso!”; 8. Usuário pressiona o botão “Ok”; 9. Sistema limpa os campos e desabilita os campos;

Fonte: Guilherme Post

Caso de uso manter cadastro de contas a pagar.

Tabela 49: Caso de Uso – Manter cadastro de contas a pagar

Nome:	Manter cadastro de contas a pagar
Objetivo:	Cadastrar as contas a serem pagas pela empresa no sistema.
Autor Primário:	Atendente.
Autor Secundário:	Não possui.
Pré-Condição:	Atendente logado no sistema e dados da conta em mãos.
Pós-Condição:	A conta estará cadastrada no sistema.
Ativação:	Ao pressionar o botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro opção Pagamento, em seguida Cadastrar Pgto; 2. Sistema apresenta a tela com os campos desabilitados; 3. Usuário pressiona o botão “Novo”; 4. Sistema habilita campos; 5. Usuário informa o número do pedido de compra da empresa; 6. Sistema consulta os dados da compra; <ul style="list-style-type: none"> 6.1 Compra não cadastrada, sistema apresenta mensagem “Compra não cadastrada!”; 6.2 Compra cadastrada, sistema apresenta dados da compra e fornecedor a qual foi solicitado a compra; 7. Usuário informa o número da nota fiscal de entrada e pressiona o botão “Salvar”; 8. Sistema apresenta mensagem “Gravação realizada com sucesso!”; 9. Usuário pressiona o botão “Ok”; 10. Sistema limpa os campos, desabilita os campos e fecha tela;

Fonte: Guilherme Post

Caso de uso consultar recebimento.

Tabela 50: Caso de Uso – Consultar recebimentos

Nome:	Consultar recebimentos
Objetivo:	Consultar as vendas solicitadas pelos clientes em um determinado período.
Ator Primário:	Atendente.
Ator Secundário:	Gerente.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Atendente/Gerente terá as informações das vendas solicitadas pelos clientes.
Ativação:	Ao pressionar o botão “Consultar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro opção Recebimento, em seguida Consultar Recebimento; 2. Sistema apresenta a tela de consulta de recebimentos com os campos desabilitados; 3. Usuário pressiona o botão “Nova”; 4. Sistema habilita campos; 5. Usuário informa os dados da consulta e pressiona o botão “Consultar”; 6. Sistema consulta os dados do recebimento; <ul style="list-style-type: none"> 6.1 Recebimento não cadastrado, sistema apresenta mensagem “Recebimento não cadastrado!”; 6.2 Recebimento cadastrado, sistema apresenta dados dos recebimentos; 7. Sistema apresenta relatório conforme filtros informados em tela.

Fonte: Guilherme Post

Caso de uso consultar contas a pagar.

Tabela 51: Caso de Uso – Consultar contas a pagar

Nome:	Consultar contas a pagar
Objetivo:	Consultar as contas a pagar feitas pela empresa no sistema.
Ator Primário:	Atendente.
Ator Secundário:	Gerente.
Pré-Condição:	Atendente/Gerente logado no sistema e dados da pesquisa que deseja solicitar.
Pós-Condição:	Apresentar ao gerente/atendente os dados dos pagamentos.
Ativação:	Ao pressionar o botão “Consultar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro opção Pagamento, em seguida Consultar Pgto; 2. Sistema apresenta a tela de consulta de pagamentos com os campos desabilitados; 3. Usuário pressiona o botão “Nova”; 4. Sistema habilita campos; 5. Usuário informa os dados da consulta; 6. Sistema consulta os dados da consulta; <ul style="list-style-type: none"> 6.1 Pagamento não cadastrado, sistema apresenta mensagem “Pagamento não cadastrado!”; 6.2 Pagamento cadastrado, sistema apresenta lista dos pagamentos; 7. Usuário poderá imprimir relatório clicando no botão “Imprimir”; 8. Sistema imprime relatório; 9. Usuário pressiona o botão “Cancelar”; 10. Sistema limpa os campos, desabilita os campos e fecha tela.

Fonte: Guilherme Post

Caso de uso manter cadastro de categoria.

Tabela 52: Caso de Uso – Manter cadastro de categoria

Nome:	Manter cadastro de categoria.
Objetivo:	Manter o cadastro de categorias dos produtos.
Ator Primário:	Atendente.
Ator Secundário:	Não possui.
Pré-Condição:	Atendente logado no sistema.
Pós-Condição:	A categoria estará cadastrada no sistema.
Ativação:	Ao pressionar no botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Suprimentos a opção Estoque, em seguida Cadastrar Categoria; 2. Sistema apresenta a tela de cadastro de categoria com os campos desabilitados; 3. Usuário pressiona o botão “Nova”; 4. Sistema habilita campos; 5. Usuário informa a descrição da categoria e pressiona no botão “Salvar”; 6. Sistema grava descrição no banco de dados, apresenta mensagem “Gravação realizada com Sucesso!”; 7. Usuário pressiona no botão “Ok”; 8. Sistema desabilita campos;

Fonte: Guilherme Post

Caso de uso manter cadastro de bairro.

Tabela 53: Caso de Uso – Manter cadastro de bairro

Nome:	Manter cadastro de bairro.
Objetivo:	Manter o cadastro de endereço.
Ator Primário:	Atendente.
Ator Secundário:	Não possui.
Pré-Condição:	Atendente logado no sistema.
Pós-Condição:	O endereço estará cadastrado no sistema.
Ativação:	Ao pressionar no botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Comercial a opção Endereço, em seguida Cadastrar Bairro; 2. Sistema apresenta a tela de cadastro de bairro com os campos desabilitados; 3. Usuário pressiona o botão “Novo”; 4. Sistema habilita campos; 5. Usuário informa a descrição do bairro, seleciona a cidade e estado, pressiona no botão “Salvar”; 6. Sistema grava descrição no banco de dados, apresenta mensagem “Gravação realizada com Sucesso!”; 7. Usuário pressiona no botão “Ok”; 8. Sistema desabilita campos;

Fonte: Guilherme Post

Caso de uso manter cadastro de vendas com cheques.

Tabela 54: Caso de Uso – Manter cadastro de vendas com cheques.

Nome:	Manter cadastro de vendas com cheques.
Objetivo:	Manter o cadastro de vendas efetuadas com cheques.
Ator Primário:	Atendente.
Ator Secundário:	Não possui.
Pré-Condição:	Atendente logado no sistema.
Pós-Condição:	Os cheques estarão cadastrados no sistema.
Ativação:	Ao pressionar no botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção cheque, em seguida cadastrar cheques; 2. Sistema apresenta a tela de cadastro de cheques; 3. Usuário pressiona o botão “Salvar”; 4. Sistema grava dado e habilita campo inferior; 5. Usuário informa os dados dos cheques e pressiona o botão “Salvar”; 6. Sistema grava os dado e apresenta “Gravação realizada com Sucesso!”; 7. Usuário pressiona no botão “Ok”; 8. Sistema desabilita campos;

Fonte: Guilherme Post

Caso de uso manter cadastro de vendas com cartão.

Tabela 55: Caso de Uso – Manter cadastro de vendas com cartão.

Nome:	Manter cadastro de vendas com cartão.
Objetivo:	Manter o cadastro de vendas efetuadas com cartão.
Ator Primário:	Atendente.
Ator Secundário:	Não possui.
Pré-Condição:	Atendente logado no sistema.
Pós-Condição:	As vendas com cartão estarão cadastradas no sistema.
Ativação:	Ao pressionar no botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção cartão, em seguida cadastrar vendas com cartão; 2. Sistema apresenta a tela de cadastro de vendas com cartão, com os campos desabilitados; 3. Usuário pressiona o botão “Novo”; 4. Sistema habilita campos; 5. Usuário informa os dados do cliente, venda e cartão e pressiona o botão “Salvar”; 6. Sistema grava os dados e apresenta mensagem “Gravação realizada com sucesso!”; 7. Usuário pressiona no botão “Ok”; 8. Sistema desabilita campos e limpa campos;

Fonte: Guilherme Post

Caso de uso manter cadastro de contas.

Tabela 56: Caso de Uso – Manter cadastro de contas.

Nome:	Manter cadastro de contas.
Objetivo:	Manter o cadastro de contas da empresa.
Ator Primário:	Gerente.
Ator Secundário:	Não possui.
Pré-Condição:	Gerente logado no sistema.
Pós-Condição:	A conta estará cadastrada no sistema.
Ativação:	Ao pressionar no botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção conta, em seguida cadastrar conta; 2. Sistema apresenta a tela de cadastro de conta, com os campos desabilitados; 3. Usuário pressiona o botão “Novo”; 4. Sistema habilita campos; 5. Usuário informa o nome da conta e pressiona o botão “Salvar”; 6. Sistema grava os dados e apresenta mensagem “Gravação realizada com sucesso!”; 7. Usuário pressiona no botão “Ok”; 8. Sistema desabilita campos e limpa campos;

Fonte: Guilherme Post

Caso de uso manter cadastro de cartão.

Tabela 57: Caso de Uso – Manter cadastro de cartão.

Nome:	Manter cadastro de cartão.
Objetivo:	Manter o cadastro de cartão das financeiras contratadas pela empresa.
Ator Primário:	Gerente.
Ator Secundário:	Não possui.
Pré-Condição:	Gerente logado no sistema.
Pós-Condição:	O cartão estará cadastrada no sistema.
Ativação:	Ao pressionar no botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção cartão, em seguida cadastrar cartão; 2. Sistema apresenta a tela de cadastro de cartão, com os campos desabilitados; 3. Usuário pressiona o botão “Novo”; 4. Sistema habilita campos; 5. Usuário informa o nome da cartão e pressiona o botão “Salvar”; 6. Sistema grava os dados e apresenta mensagem “Gravação realizada com sucesso!”; 7. Usuário pressiona no botão “Ok”; 8. Sistema desabilita campos e limpa campos;

Fonte: Guilherme Post

Caso de uso manter estoque.

Tabela 58: Caso de Uso – Manter cadastro de cartão.

Nome:	Manter estoque.
Objetivo:	Manter o controle da situação dos produtos no estoque, podendo realizar reajustes nos preços dos produtos.
Autor Primário:	Atendente.
Autor Secundário:	Não possui.
Pré-Condição:	Gerente logado no sistema.
Pós-Condição:	Os produtos estarão atualizados no sistema.
Ativação:	Ao pressionar no botão “Salvar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Suprimentos a opção estoque, em seguida consultar produto e pressionar o botão “Reajuste” na parte inferior da tela de consulta de produtos; 2. Sistema apresenta a tela de controle de estoque; 3. Usuário poderá selecionar os itens por categoria, produto, opção todos os produtos, e inserir os dados da alteração de preço e pressiona o botão “Salvar”; 4. Sistema grava os dados e apresenta mensagem “Alteração realizada com sucesso!”; 5. Usuário pressiona no botão “Ok”; 6. Sistema limpa os campos;

Fonte: Guilherme Post

Caso de uso controlar vendas feitas com cheques.

Tabela 59: Caso de Uso – Controlar vendas feitas com cheques.

Nome:	Controlar vendas feitas com cheques.
Objetivo:	Manter o controle das vendas feitas com cheques.
Autor Primário:	Atendente
Autor Secundário:	Gerente.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Apresentar os dados das vendas feitas com cheques ao Atendente/Gerente
Ativação:	Ao pressionar no botão “Consultar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção cheque, em seguida controle de cheques; 2. Sistema apresenta a tela de controle de cheques; 3. Usuário poderá selecionar as opções em tela para visualizar o dados das vendas feitas com cheques e pressiona o botão “Consultar”; 4. Sistema consulta as vendas conforme parâmetros informados em tela; 5. Sistema apresenta os dados das vendas feitas com cheques.

Fonte: Guilherme Post

Caso de uso consultar fluxo de caixa.

Tabela 60: Caso de Uso – Consultar fluxo de caixa.

Nome:	Consultar fluxo de caixa.
Objetivo:	Consultar o fluxo de caixa da empresa em um determinado intervalo de tempo.
Ator Primário:	Gerente.
Ator Secundário:	Atendente.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Atendente/Gerente obterão as informações de saldo em caixa da empresa.
Ativação:	Ao pressionar no botão “Consultar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção caixa, em seguida consultar fluxo de caixa; 2. Sistema apresenta a tela de consulta de fluxo de caixa; 3. Usuário seleciona contas a pagar ou contas a receber; 4. Usuário poderá informar o período que deseja consultar e os parâmetros da pesquisa; 5. Usuário pressiona o botão consultar; 6. Sistema apresenta todas as movimentações <ol style="list-style-type: none"> 6.1 Sistema apresenta os dados das contas a pagar: <ol style="list-style-type: none"> 6.1.1 Exibe o numero da ordem de compra; 6.1.2 Exibe fornecedor a ser pago; 6.1.3 Exibe data de vencimento; 6.1.4 Exibe dias em atraso; 6.1.5 Exibe a forma de pagamento; 6.1.6 Exibe a condição de pagamento; 6.1.7 Exibe o banco a ser pago; 6.1.8 Exibe o valor total da parcela; 6.1.9 Exibe o status da conta (aberto ou encerrado) 6.2 Sistema apresenta dados das contas a receber: <ol style="list-style-type: none"> 6.2.1 Exibe o numero do pedido de venda; 6.2.2 Exibe o nome do cliente; 6.2.3 Exibe o status do da venda (aberto ou encerrado); 6.2.4 Exibe data de vencimento; 6.2.5 Exibe dias em atraso; 6.2.6 Exibe forma de pagamento; 6.2.7 Exibe a condição de pagamento; 6.2.8 Exibe o valor total da parcela;

Fonte: Guilherme Post

Caso de uso controlar vendas feitas com cartão.

Tabela 61: Caso de Uso – Controlar vendas feitas com cartão.

Nome:	Controlar vendas feitas com cartão.
Objetivo:	Manter o controle das vendas feitas com cartão.
Ator Primário:	Atendente.
Ator Secundário:	Gerente.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Apresentar os dados das vendas feitas com cartão ao Atendente/Gerente
Ativação:	Ao pressionar no botão “Consultar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção cartão, em seguida controle de cartão; 2. Sistema apresenta a tela de controle de cartão; 3. Usuário poderá selecionar as opções em tela para visualizar o dados das vendas feitas com cartão e pressiona o botão “Consultar”; 4. Sistema consulta as vendas conforme parâmetros informados em tela; 5. Sistema apresenta os dados das vendas feitas com cheques.

Fonte: Guilherme Post

Caso de uso gerar relatório de contas a receber.

Tabela 62: Caso de Uso – Gerar relatório de contas a receber

Nome:	Gerar relatório de contas a receber.
Objetivo:	Apresentar o relatório no período desejado das contas a receber.
Ator Primário:	Atendente.
Ator Secundário:	Gerente.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Apresentar os dados das contas a receber gerada pelas vendas feitas aos clientes.
Ativação:	Ao pressionar no botão “Consultar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção recebimento; 2. Sistema apresenta a tela de consulta a recebimentos; 3. Usuário deverá informar os filtros que deseja utilizar como parâmetros para listar o relatório; 4. Sistema consulta as contas a receber conforme parâmetros informados pelo usuário; 5. Sistema apresenta os dados das contas a receber através do relatório.

Fonte: Guilherme Post

Caso de uso gerar relatório de contas a pagar.

Tabela 63: Caso de Uso – Gerar relatório de contas a pagar

Nome:	Gerar relatório de contas a pagar.
Objetivo:	Apresentar o relatório no período desejado das contas a pagar.
Ator Primário:	Atendente.
Ator Secundário:	Gerente.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Apresentar os dados das contas a receber gerada pelas compras feitas aos fornecedores.
Ativação:	Ao pressionar no botão “Consultar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção pagamento. 2. Sistema apresenta a tela de consulta a pagamentos; 3. Usuário deverá informar os filtros que deseja utilizar como parâmetros para listar o relatório; 4. Sistema consulta as contas a pagar conforme parâmetros informados pelo usuário; 5. Sistema apresenta os dados das contas a pagar através do relatório.

Fonte: Guilherme Post

Caso de uso gerar relatório de produtos em falta no estoque.

Tabela 64: Caso de Uso – Gerar relatório de produtos em falta no estoque

Nome:	Gerar relatório de produtos em falta no estoque.
Objetivo:	Listar os produtos que estão em falta no estoque ou abaixo do mínimo.
Ator Primário:	Atendente.
Ator Secundário:	Gerente.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Apresentar os itens em falta ou abaixo do mínimo ao Atendente/Gerente
Ativação:	Ao pressionar no botão “Consultar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Suprimentos a opção produto, consultar produto; 2. Sistema apresenta a tela de consulta de produtos; 3. Usuário deverá pressionar o botão relatório; 4. Sistema apresenta tela para listar os parâmetros da pesquisa; 5. Usuário informa os filtros que deseja utilizar como parâmetros para listar o relatório; 6. Sistema consulta os produtos conforme parâmetros informados pelo usuário; 7. Sistema apresenta os dados das contas a pagar através do relatório.

Fonte: Guilherme Post

Caso de uso gerar relatório de fluxo de caixa.

Tabela 65: Caso de Uso – Gerar relatório de fluxo de caixa.

Nome:	Gerar relatório de fluxo de caixa.
Objetivo:	Apresentar os dados das contas a pagar e a receber.
Ator Primário:	Atendente.
Ator Secundário:	Gerente.
Pré-Condição:	Atendente/Gerente logado no sistema.
Pós-Condição:	Apresentar os dados das vendas e compras.
Ativação:	Ao pressionar no botão “Consultar”.
Fluxo principal:	<ol style="list-style-type: none"> 1. Usuário seleciona na tela principal o menu Financeiro a opção conta, em seguida controle de conta; 2. Sistema apresenta a tela de controle de conta; 3. Usuário poderá selecionar as opções em tela para visualizar as contas a receber e a pagar; 4. Usuário poderá listar o relatório por conta a receber, a pagar, tipo de conta; 5. Usuário deverá pressionar o botão “Imprimir”; 6. Sistema consulta as contas a pagar e receber conforme parâmetros informados em tela; 7. Sistema apresenta os dados através do relatório.

Fonte: Guilherme Post

3.5. Análise de Requisitos

A partir desta etapa que serão analisadas as funções, desempenho que o sistema deverá realizar assim como necessidades dos clientes, se foram bem compreendidas nas etapas anteriores.

3.5.1. Analisar Arquitetura

Atualmente a empresa não disponibiliza de nenhum recurso tecnológico para gerenciar os processos realizados pela empresa. Os processos são feitos manualmente, as anotações são feitas em papeis, agendas e os cálculos manualmente.

Como o projeto a ser desenvolvido não possui um escopo muito detalhado, com funcionalidades simples, não será abordada a análise da arquitetura.

3.5.2. Analisar Caso de Uso

A análise de caso de uso detalha a troca de mensagens entre as classes de entidades, interface e controle em uma linha de tempo. Na tabela 66 é apresentado às entidades, controles e interfaces usadas durante a execução dos casos de uso.

Analizar caso de uso Efetuar *Login*:

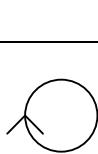
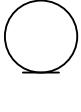
Tabela 66: Analisar Casos de Uso – Efetuar login.

	frmMenuInicial	O objetivo desta classe é apresentar uma interface gráfica, permitindo que o usuário entre com os dados referentes ao funcionário, como, login e senha.
	controlAcesso	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	SenhaEntity	O objetivo desta classe é de fornecer a senha e login cadastrado para cada funcionário.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Funcionário:

Tabela 67: Analisar Casos de Uso – Manter cadastro de funcionário.

	frmCadFuncionário	O objetivo desta classe é apresentar uma interface gráfica, permitindo que o usuário entre com os dados referentes ao funcionário, como, nome endereço, telefone, data admissão, cargo e outros.
	controlFuncionário	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	FuncionárioEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Cliente:

Tabela 68: Analisar Casos de Uso – Manter cadastro de cliente.

	frmCadCliente	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes ao cliente, como, nome endereço, telefone e outros.
	controlCliente	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ClienteEntity	O objetivo é ser alimentada com os dados da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Produto:

Tabela 69: Analisar Casos de Uso – Manter cadastro de produto.

	frmCadProduto	O objetivo desta classe é apresentar uma interface gráfica, permitindo que o usuário entre com os dados referentes ao produto, como, descrição do produto, unidade de medida, preço de custo, margem de lucro.
	controlProduto	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ProdutoEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter cadastro de Cartão:

Tabela 70: Analisar Casos de Uso – Manter cadastro de cartão.

	frmCadCartao	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes ao cartão, como, a descrição do cartão.
	controlCartao	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	CartaoEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Compra:

Tabela 71: Analisar Casos de Uso – Manter cadastro de compra.

	frmCadCompra	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes à compra, como, razão social, CNPJ do fornecedor, condição de pagamento, forma de pagamento, itens da compra.
	controlCompra	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	CompraEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Condição de Pagamento:

Tabela 72: Analisar Casos de Uso – Manter cadastro de condição de pagamento.

	frmCadCondPgto	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário informe apenas a descrição.
	controlCondPgto	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	CondPgtoEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Forma de Pagamento:

Tabela 73: Analisar Casos de Uso – Manter cadastro de forma de pagamento.

	frmCadFormaPgto	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes à forma de pagamento, usuário informa apenas a descrição.
	controlFormaPgto	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	FormaPgtoEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Bairro:

Tabela 74: Analisar Casos de Uso – Manter cadastro de bairro.

	frmCadBairroPgto	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes ao bairro, como, descrição do bairro, cidade e estado.
	controlBairroPgto	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	FormaBairroEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Categoria:

Tabela 75: Analisar Casos de Uso – Manter cadastro categoria.

	frmCadCategoria	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes à Categoria, usuário informa apenas a descrição.
	controlCategoria	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	CategoriaEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Contas a Pagar:

Tabela 76: Analisar Casos de Uso – Manter cadastro de contas a pagar.

	frmCadContaPagar	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes às Contas a Pagar, como, numero do pedido de compra, numero da nota fiscal de entrada, valor com impostos
	controlContaPagar	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ContaPagarEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Contas a Receber:

Tabela 77: Analisar Casos de Uso – Manter cadastro de contas a receber.

	frmCadContaReceber	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes às Contas a Receber, como, nome do cliente, CPF, data do pagamento, valor do pagamento.
	controlContaReceber	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ContaReceberEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Controle de Estoque:

Tabela 78: Analisar Casos de Uso – Manter controle de estoque.

	frmCadEstoque	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes aos produtos, como, margem de lucro, situação.
	controlEstoque	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	EstoqueEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter cadastro de senhas:

Tabela 79: Analisar Casos de Uso – Manter cadastro de senhas.

	frmCadSenha	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes à Senhas do usuário, como, o funcionário, ações sobre o que pode fazer (salvar, excluir, consultar, editar), e telas que pode acessar.
	controlSenha	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	SenhaEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter cadastro de contas:

Tabela 80: Analisar Casos de Uso – Manter cadastro de conta.

	frmCadConta	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário informe a descrição da conta.
	controlConta	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ContaEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Venda com Cartão:

Tabela 81: Analisar Casos de Uso – Manter venda com cartão.

	frmCadVendaCartao	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes à venda cartão, como, tipo de cartão, numero do cartão, nome cliente, validade, código autorização, numero de parcelas, valor, conta a ser lançada.
	controlVendaCartao	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	VendaCartaoEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Consultar Fluxo de Caixa:

Tabela 82: Analisar Casos de Uso – Consultar fluxo de caixa.

	frmConsFluxo	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário visualize os dados referentes vendas e compras, como, data de entrada ou saída, numero do documento, valor, fornecedor ou cliente.
	controlConsFluxo	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	VendaEntity	O objetivo desta classe é fornecer os dados armazenados no banco.

Fonte: Guilherme Post

Análise do caso de uso Manter Venda com Cheque:

Tabela 83: Analisar Casos de Uso – Manter venda com cheque.

	frmCadVendaCheque	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes à venda feitas com cheque, como, numero de cheques, valor total, nome cliente, valor do cheque, banco, agência, numero da conta.
	controlVendaCheque	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	VendaCheque Entity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Controlar Vendas com Cheques:

Tabela 84: Analisar Casos de Uso – Controlar vendas com cheques.

	frmConsVendaCheque	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário visualize os dados referentes venda feitas com cheques, como, cliente, data de emissão, valor, nº cheque, banco.
	controlVendaCheque	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	VendaEntity	O objetivo desta classe é fornecer os dados armazenados no banco referente às vendas emitidas aos clientes.

Fonte: Guilherme Post

Análise do caso de uso Consultar Venda:

Tabela 85: Analisar Casos de Uso – Consultar venda.

	frmConsVenda	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário visualize os dados referentes a venda, como, nome do cliente, numero do pedido, data de emissão, condição de pagamento, forma de pagamento, valor total e itens do pedido.
	controlVenda	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	VendaEntity	O objetivo desta classe é fornecer os dados armazenados no banco referente às vendas emitidas aos clientes.

Fonte: Guilherme Post

Análise do caso de uso Consultar Pagamento:

Tabela 86: Analisar Casos de Uso – Consultar contas a pagar.

	frmConsPgto	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário visualize os dados referentes ás contas a pagar, como, nome do fornecedor, numero do pedido, data de emissão, condição de pagamento, valor total e data vencimento.
	controlPgto	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ContaPagarEntity	O objetivo desta classe é fornecer os dados armazenados no banco referente ás compras emitidas aos fornecedores.

Fonte: Guilherme Post

Análise do caso de uso Consultar Recebimento:

Tabela 87: Analisar Casos de Uso – Consultar contas a receber.

	frmConsRecebimento	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário visualize os dados referentes às contas a receber, como, nome do cliente, numero do pedido, data de emissão, condição de pagamento, forma de pagamento, valor total e data do pagamento.
	controlContaReceb	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ContaReceberEntity	O objetivo desta classe é fornecer os dados armazenados no banco referente às vendas emitidas aos clientes.

Fonte: Guilherme Post

Análise do caso de uso Consultar Compras:

Tabela 88: Analisar Casos de Uso – Consultar compras.

	frmConsCompra	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário visualize os dados referentes à compra, como, fornecedor, numero pedido, data de emissão, condição de pagamento, valor total e itens.
	controlCompra	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	CompraEntity	O objetivo desta classe é fornecer os dados armazenados no banco referente as compra emitidas aos fornecedores.

Fonte: Guilherme Post

Análise do caso de uso Consultar Clientes:

Tabela 89: Analisar Casos de Uso – Consultar cliente.

	frmConsCliente	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário visualize os dados referentes ao cliente, como, nome do cliente, endereço, telefone e e-mail.
	controlCliente	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ClienteEntity	O objetivo desta classe é fornecer os dados armazenados no banco referente aos clientes.

Fonte: Guilherme Post

Análise do caso de uso Consultar Produtos:

Tabela 90: Analisar Casos de Uso – Consultar produto.

	frmConsProduto	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário visualize os dados referentes ao produto, como, descrição, preço unitário, quantidade atual, quantidade mínima e categoria.
	controlProduto	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ProdutoEntity	O objetivo desta classe é fornecer os dados armazenados no banco referente aos produtos.

Fonte: Guilherme Post

Análise do caso de uso Controlar Vendas com Cartão:

Tabela 91: Analisar Casos de Uso – Controlar vendas com cartão.

	frmConsVendaCartao	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário visualize os dados referentes venda, como, cliente, data de emissão, valor da venda, data compensação, numero do cartão, conta para lançamento.
	controlVendaCartao	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	VendaCartaoEntity	O objetivo desta classe é fornecer os dados armazenados no banco referente as vendas.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Nota Fiscal de Entrada:

Tabela 92: Analisar Casos de Uso – Manter cadastro de nota fiscal de entrada.

	frmCadNFEntrada	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes à nota fiscal de entrada, como, numero do pedido de compra, itens da nota e preço unitário, quantidade de itens, valor total.
	controlNFEntrada	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	CompraEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Analizar caso de uso gerar Relatório de Contas a Receber:

Tabela 93: Analisar Casos de Uso – Gerar relatório de contas a receber.

	frmConsRecebimento	O objetivo desta classe é apresentar uma interface gráfica, permitindo que o usuário entre com os dados referentes as vendas emitidas aos clientes, como, data de emissão, forma de pagamento, condição de pagamento, data do vencimento, dias em atraso, valor total da venda, data do pagamento e status da venda (pago ou aberto).
	controlConsRecebimento	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ContaReceberEntity	Com a invocação da tela de recebimento esta classe fica responsável por fornecer os dados da venda nelas armazenada.

Fonte: Guilherme Post

Analizar caso de uso gerar Relatório de Contas a Pagar:

Tabela 94: Analisar Casos de Uso – Gerar relatório de contas a pagar.

	frmConsContaPagar	O objetivo desta classe é apresentar uma interface gráfica, permitindo que o usuário entre com os dados referentes às compras solicitadas aos fornecedores, como, data de emissão, forma de pagamento, condição de pagamento, data do vencimento, dias em atraso, valor total da compra, data do pagamento e status da compra (pago ou aberto).
	controlConsContaPagar	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ContaPagarEntity	Com a invocação da tela de consulta a contas a pagar esta classe fica responsável por fornecer os dados das compras nelas armazenada.

Fonte: Guilherme Post

Analizar caso de uso gerar Relatório de Produtos em falta no Estoque:

Tabela 95: Analisar Casos de Uso – Gerar relatório de produtos em falta no estoque.

	frmConsProduto	O objetivo desta classe é apresentar uma interface gráfica, permitindo que o usuário entre com os dados referentes ao produtos em falta no estoque, como, código do produto, descrição, quantidade mínima, quantidade atual, fornecedor, preço de custo, data da ultima compra.
	controlConsProduto	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ProdutoEntity	Com a invocação da tela de consulta a de produtos esta classe fica responsável por fornecer os dados dos produtos nela armazenada.

Fonte: Guilherme Post

Analizar caso de uso gerar Relatório de Fluxo de Caixa:

Tabela 96: Analisar Casos de Uso – Gerar relatório de fluxo de caixa.

	frmConsFluxoCaixa	O objetivo desta classe é apresentar uma interface gráfica, permitindo que o usuário entre com os dados referentes as contas a pagar e receber, como, numero do pedido, data de emissão, data vencimento, dias em atraso, valor total, forma e condição de pagamento (das compras e vendas).
	controlConsFluxoCaixa	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	ContaPagarEntity	Com a invocação da tela de consulta a de fluxo de caixa esta classe fica responsável por fornecer os dados das contas a pagar nela armazenada.
	ContaReceberEntity	Com a invocação da tela de consulta a de fluxo de caixa esta classe fica responsável por fornecer os dados das contas a receber nela armazenada.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Fornecedor:

Tabela 97: Analisar Casos de Uso – Manter cadastro de fornecedor.

	frmCadFornecedor	O objetivo desta classe é apresentar uma interface gráfica, permitindo que o usuário entre com os dados referentes ao fornecedor, como, CNPJ, razão social, pessoa de contato, inscrição estadual, endereço, telefone, data admissão, cargo e outros.
	controlFornecedor	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	FornecedorEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

Análise do caso de uso Manter Cadastro de Venda:

Tabela 98: Analisar Casos de Uso – Manter cadastro de venda.

	frmCadVenda	O objetivo desta classe é criar uma interface gráfica, permitindo que o usuário entre com os dados referentes a venda, como, nome do cliente, CPF do cliente, condição de pagamento, forma de pagamento, itens da venda.
	controlVenda	Esta classe tem objetivo de tratar todos os eventos de tela, como, cliques em botões, seleção de campos e outros.
	VendaEntity	O objetivo desta classe é ser alimentada com os dados do banco ou da tela e servir como meio de comunicação entre a camada de apresentação e a camada de persistência.

Fonte: Guilherme Post

3.5.3. Diagrama de Seqüência

O diagrama de seqüência é um dos diagramas mais utilizados na documentação dos projetos, por apresentar de forma mais detalhada as operações que cada ator. Neste projeto foram escolhidos três casos de uso, o de manter clientes, manter vendas, manter controle de estoque. Devido a estes casos de uso alcançar um maior número de operações a serem executadas para finalizar a atividade.

Diagrama de seqüência manter cadastro de cliente é apresentado na figura 19:

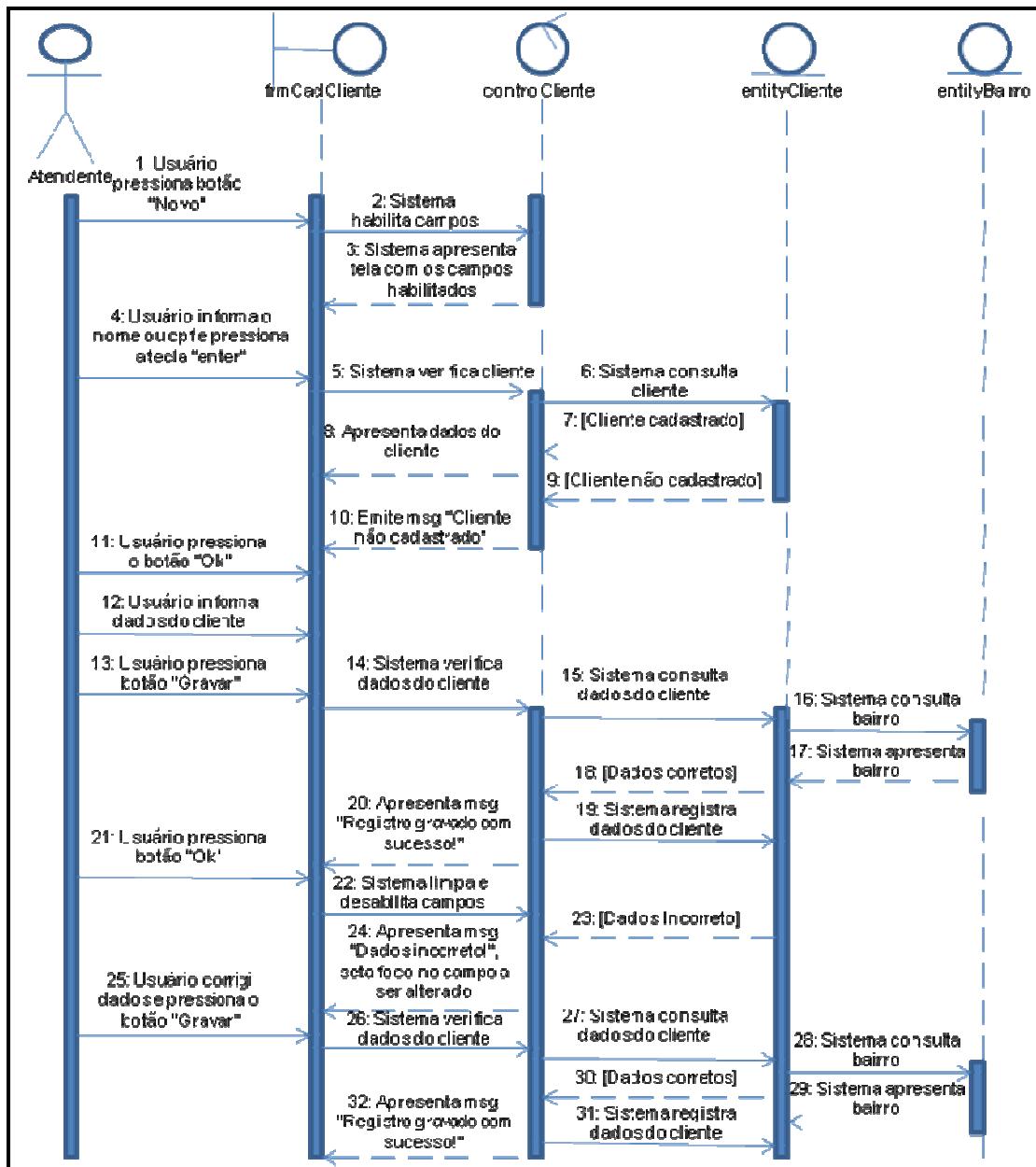


Figura 20: Diagrama de Seqüência – Manter cadastro de clientes.

Fonte: Guilherme Post

Diagrama de seqüência manter venda é apresentado na figura 20:

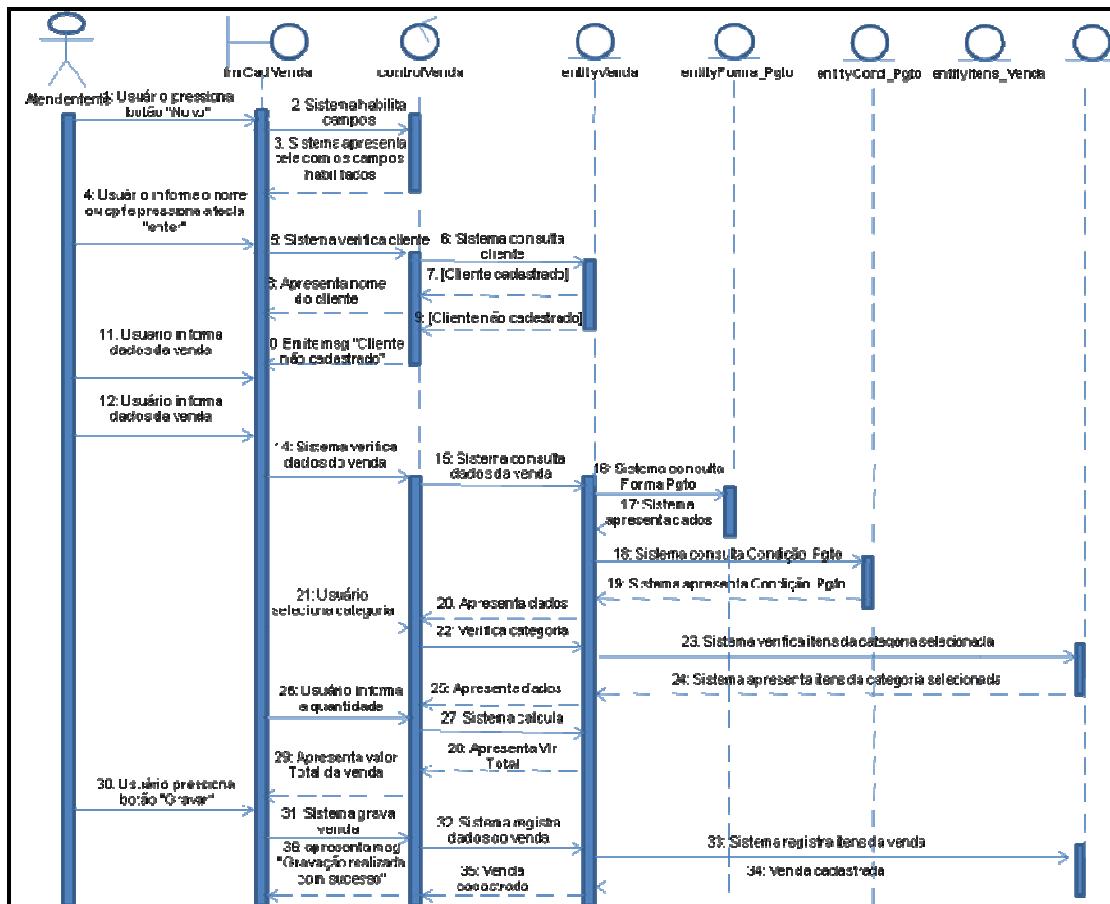


Figura 21: Diagrama de Seqüência – Manter venda.

Fonte: Guilherme Post

Diagrama de seqüência manter estoque é apresentado na figura 21:

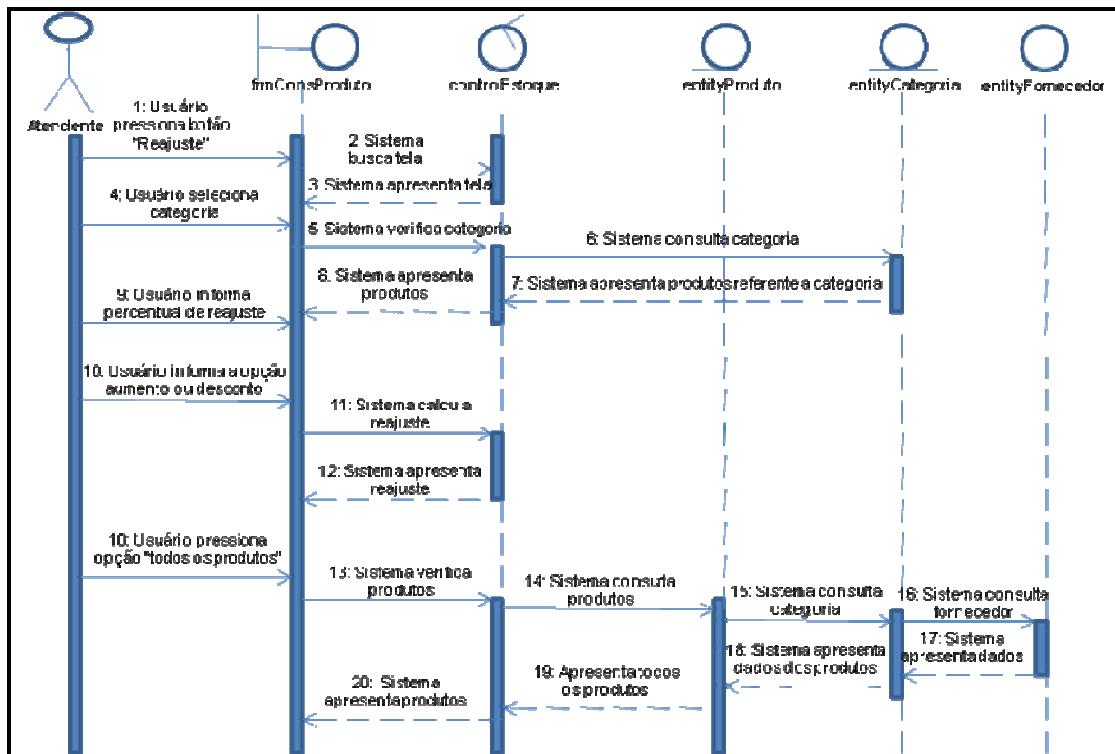


Figura 22: Diagrama de Seqüência – Manter estoque.

Fonte: Guilherme Post

3.5.4. Diagrama de Colaboração

Os diagramas de colaboração apresentam de uma forma clara e objetiva a interação do usuário com as telas (Frm), as Frm com as *Control* e a ligação com as tabelas no banco de dados (*Entity*). Foi escolhido o como segundo diagrama o de colaboração, por apresenta uma forma mais legível a ação e funções do sistema, destacando que para processos mais simples (como de cadastros) se torna mais fácil o entendimento da funcionalidade do sistema através deste diagrama.

Manter cadastro de fornecedor.

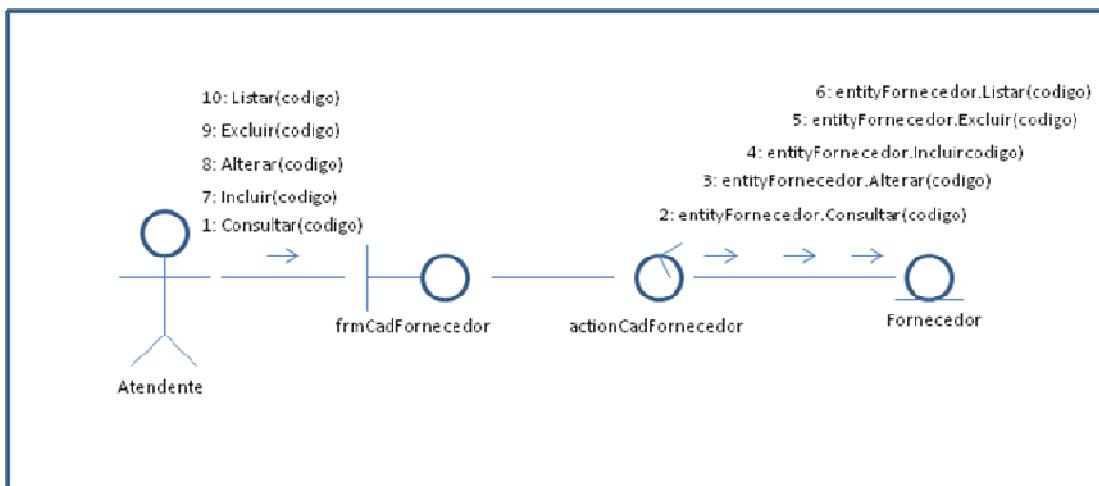


Figura 23: Diagrama de Colaboração – Manter cadastro de Fornecedor.
Fonte: Guilherme Post

Manter cadastro de Funcionário.

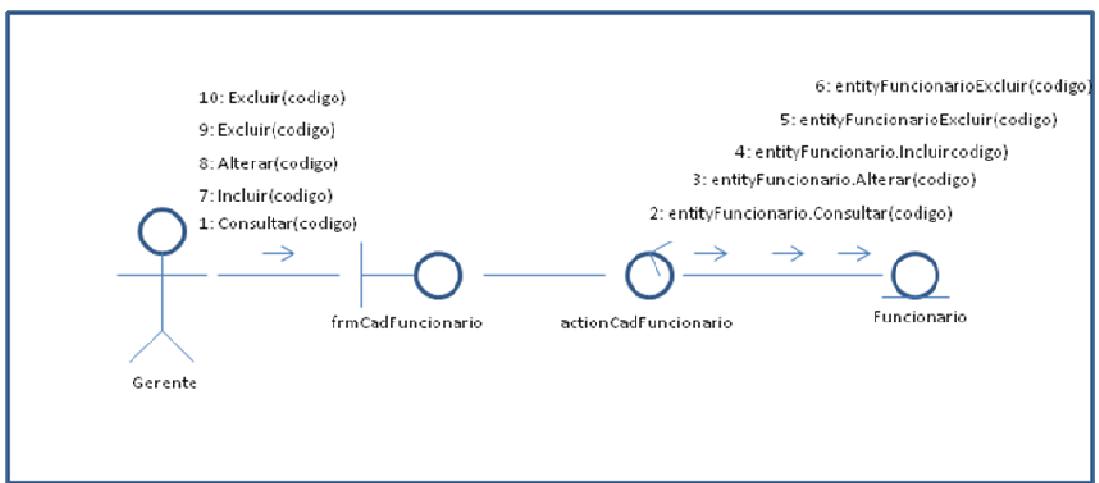


Figura 24: Diagrama de Colaboração – Manter cadastro de Funcionário.
Fonte: Guilherme Post

Manter cadastro de Senha.

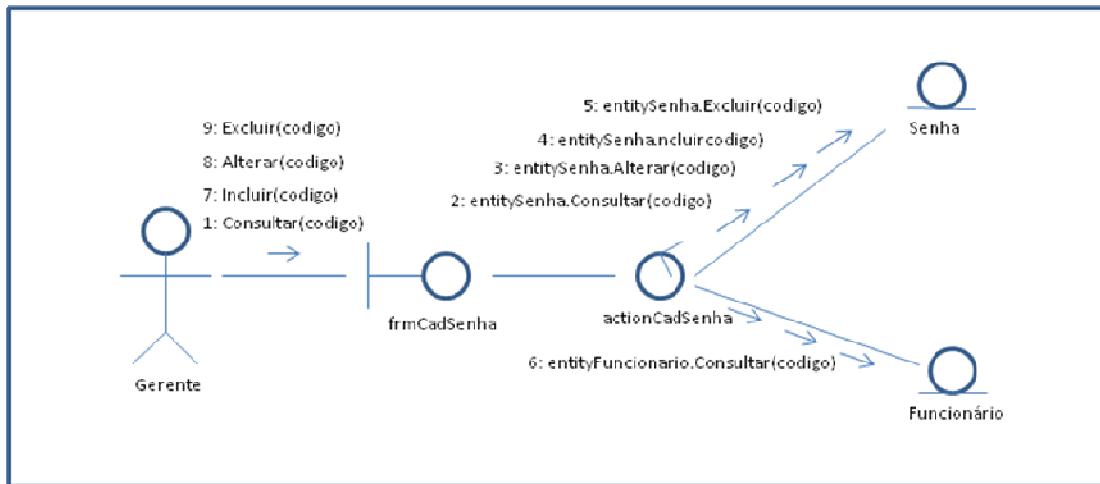


Figura 25: Diagrama de Colaboração – Manter cadastro de Senha.

Fonte: Guilherme Post

Manter cadastro de Condição de Pagamento.

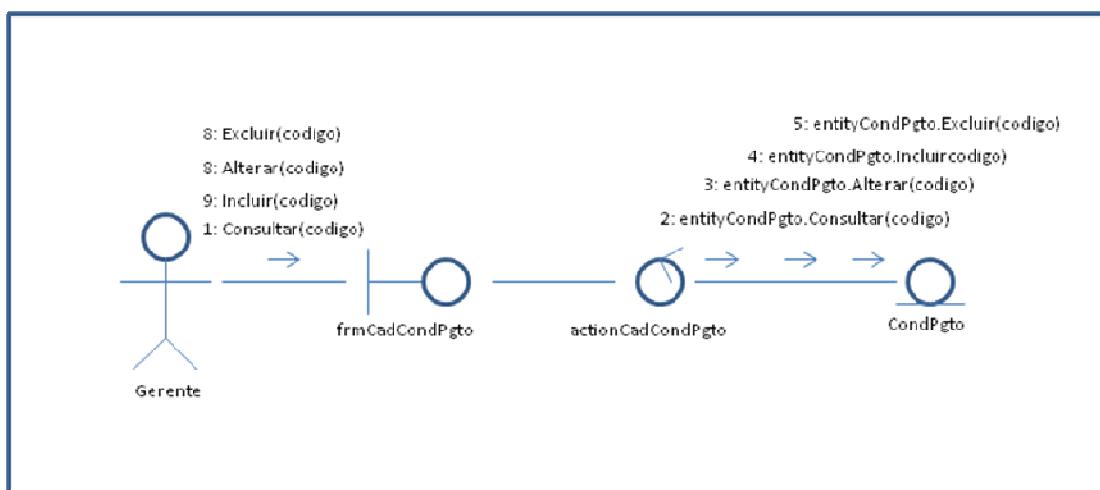


Figura 26: Diagrama de Colaboração – Manter cadastro de Condição de Pagamento.

Fonte: Guilherme Post

Manter cadastro de Forma de Pagamento.

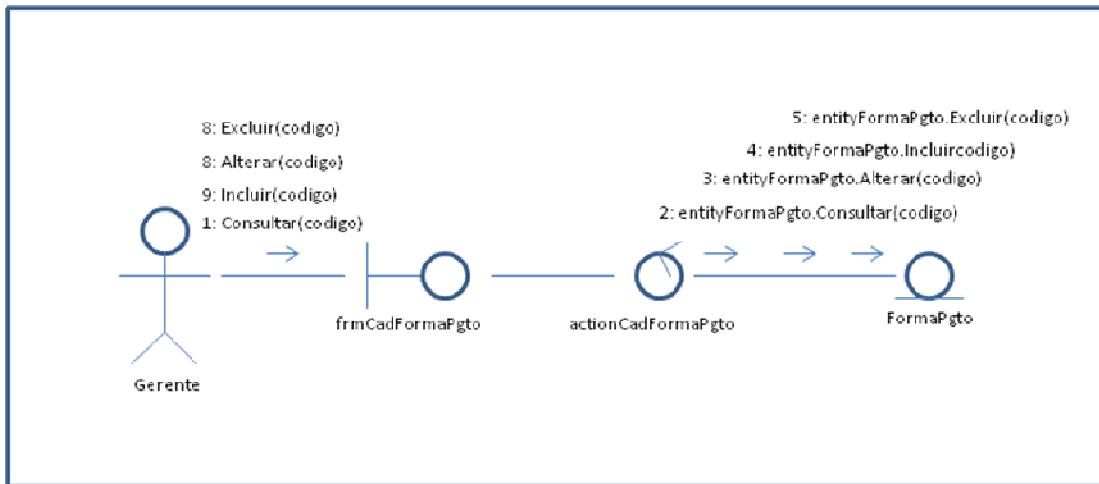


Figura 27: Diagrama de Colaboração – Manter cadastro de Forma de Pagamento.

Fonte: Guilherme Post

Manter cadastro de Categoria.

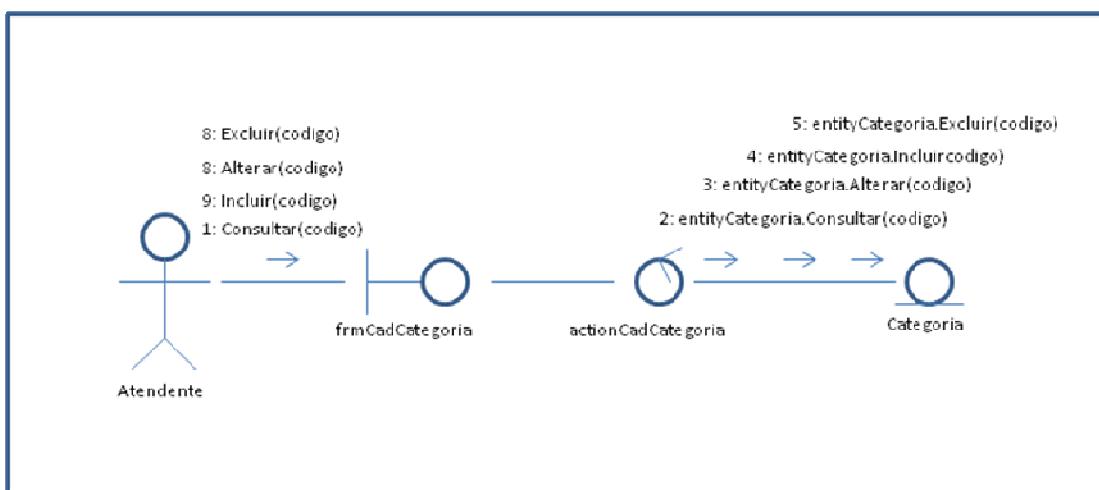


Figura 28: Diagrama de Colaboração – Manter cadastro de Categoria.

Fonte: Guilherme Post

Manter cadastro de Cartão.

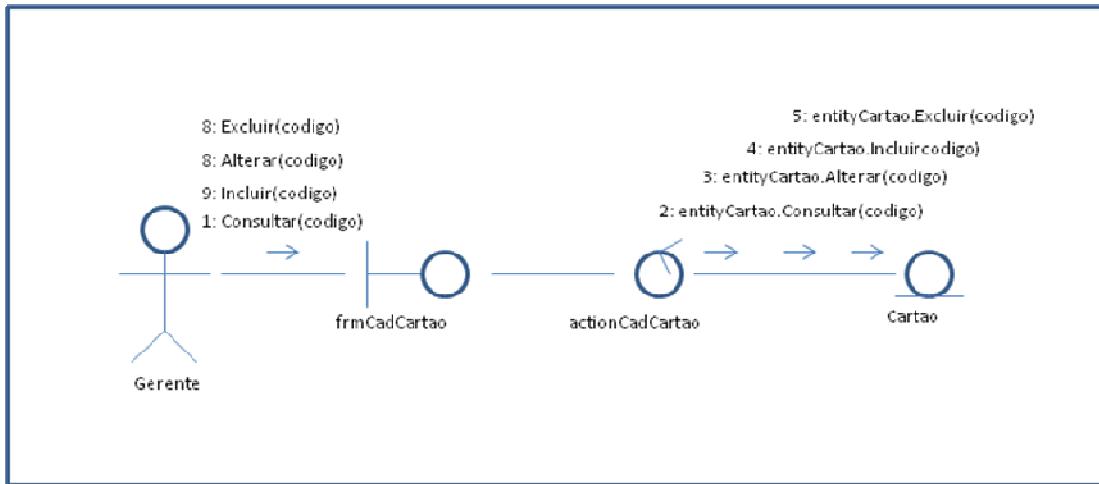


Figura 29: Diagrama de Colaboração – Manter cadastro de cartão.

Fonte: Guilherme Post

Manter cadastro de conta.

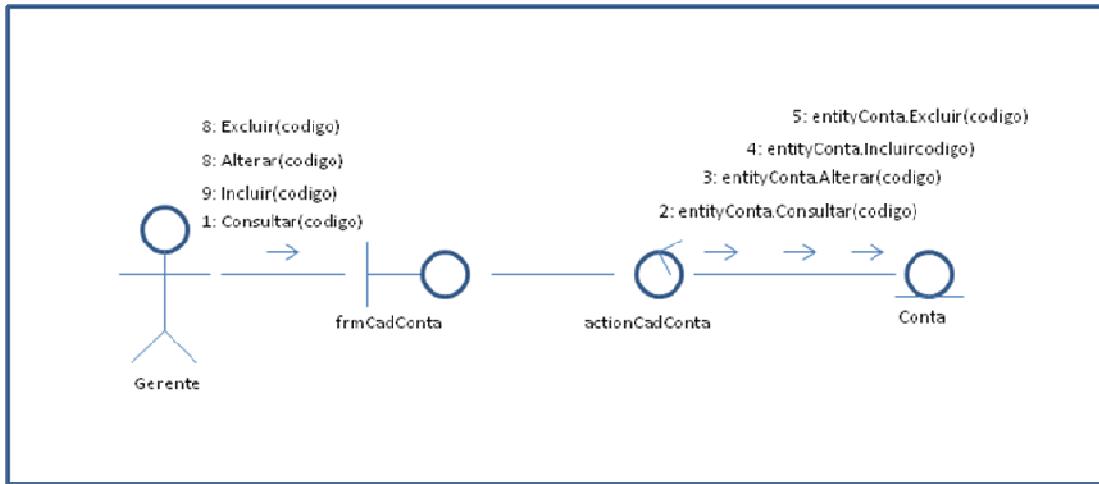


Figura 30: Diagrama de Colaboração – Manter cadastro de conta.

Fonte: Guilherme Post

Manter cadastro de cheque.

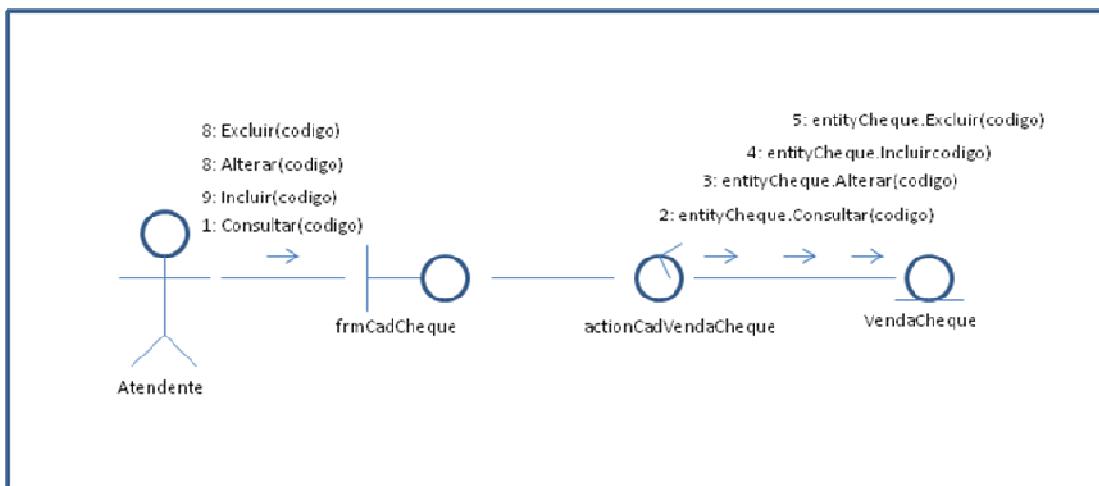


Figura 31: Diagrama de Colaboração – Manter cadastro de cheque.
Fonte: Guilherme Post

Manter cadastro de bairro.

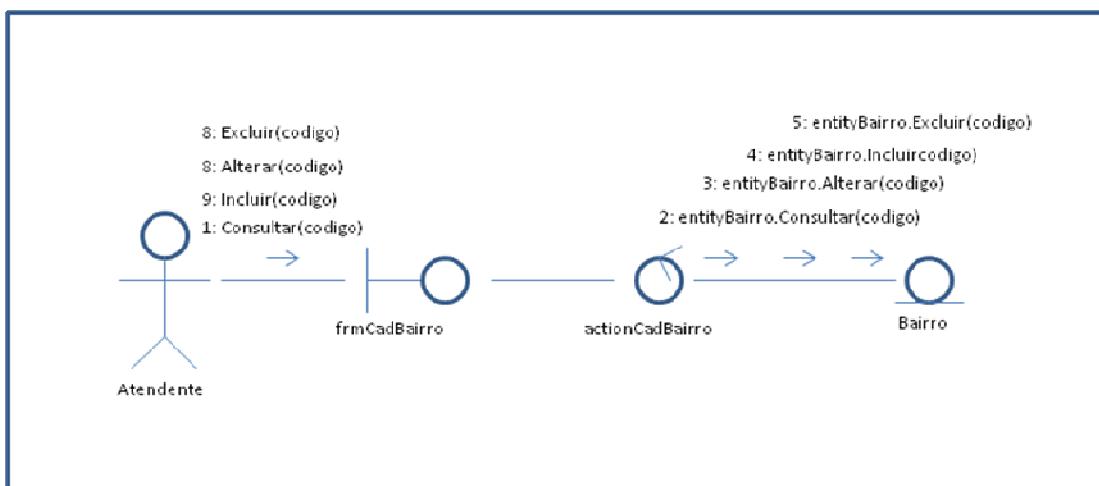


Figura 32: Diagrama de Colaboração – Manter cadastro de bairro.
Fonte: Guilherme Post

Manter cadastro de contas a pagar.

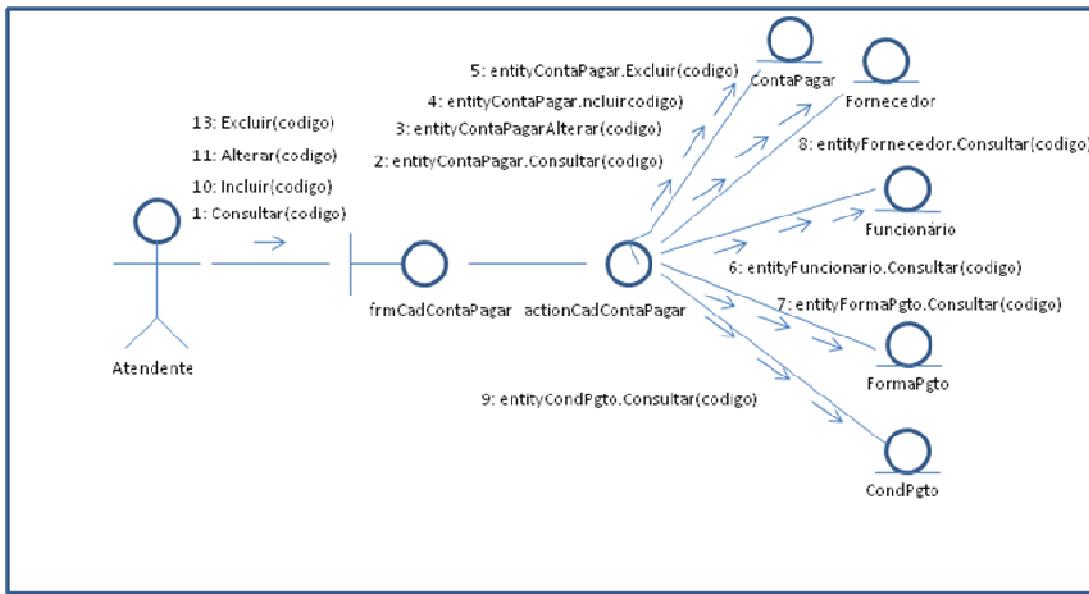


Figura 33: Diagrama de Colaboração – Manter cadastro de contas a pagar.

Fonte: Guilherme Post

Manter cadastro de contas a receber.

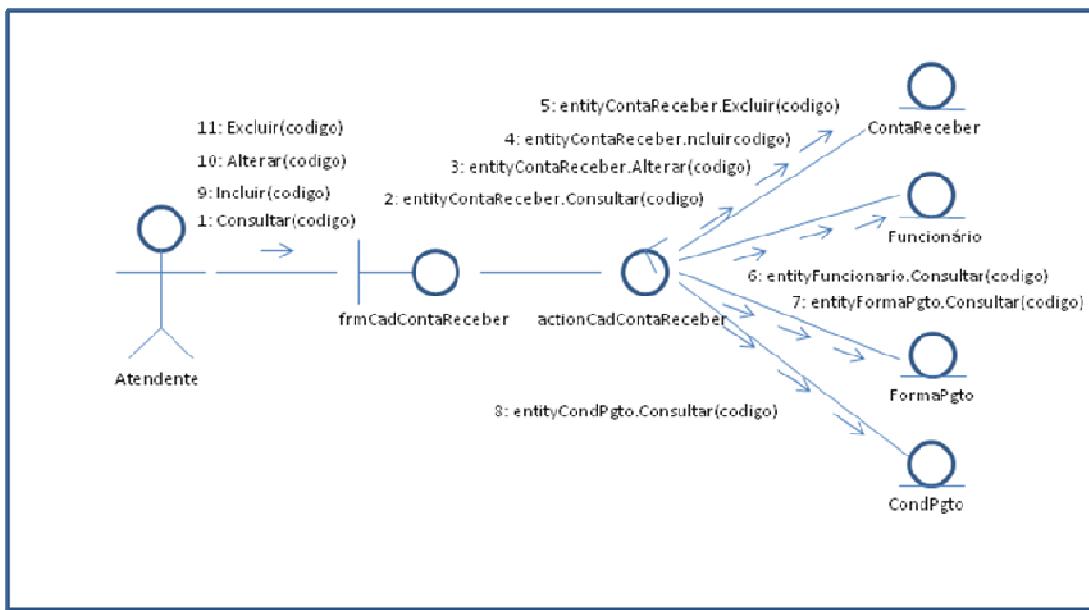


Figura 34: Diagrama de Colaboração – Manter cadastro de contas a receber.

Fonte: Guilherme Post

Manter cadastro de compras.

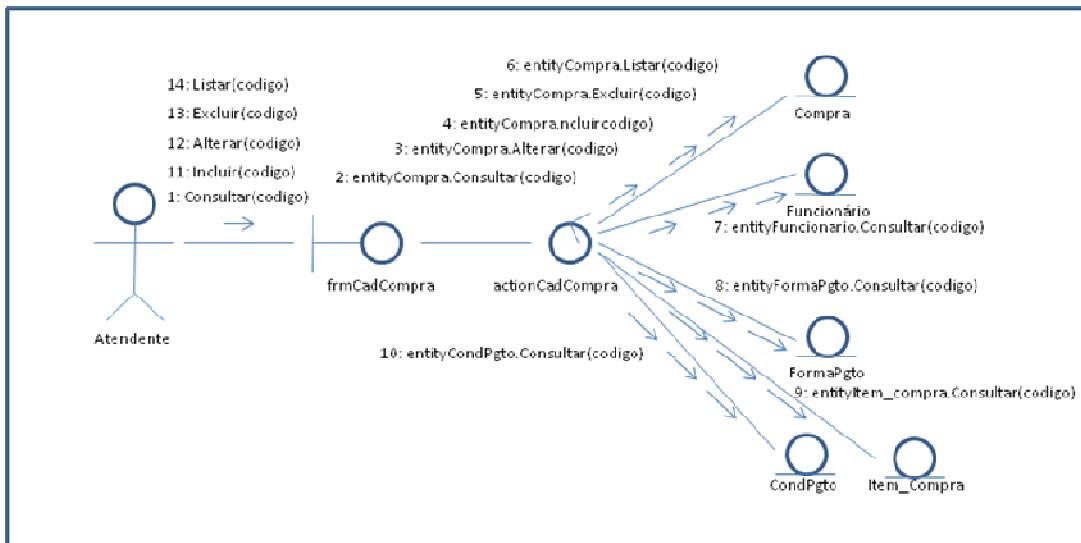


Figura 35: Diagrama de Colaboração – Manter cadastro de compras.

Fonte: Guilherme Post

Manter controle de vendas feitas com cartão.

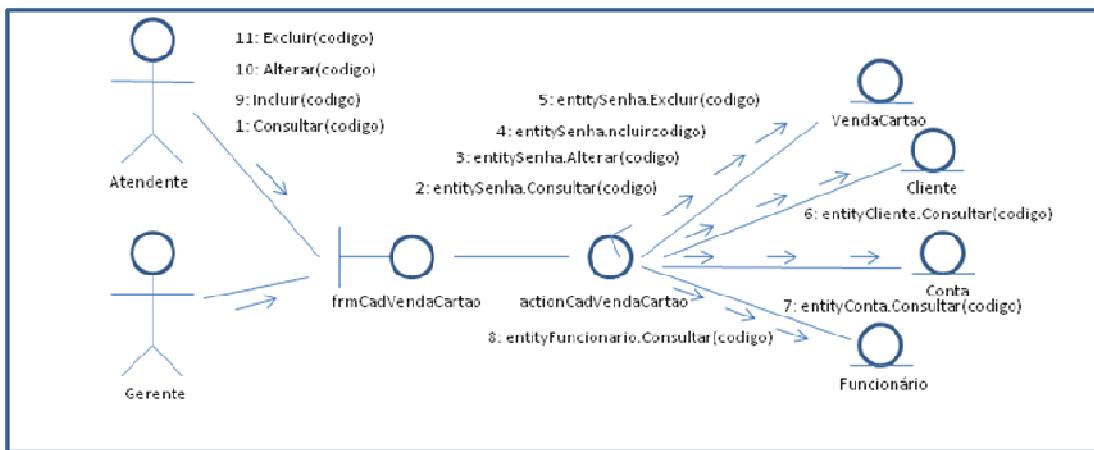


Figura 36: Diagrama de Colaboração – Manter controle de vendas feitas com cartão.

Fonte: Guilherme Post

Manter cadastro de nota fiscal de entrada.

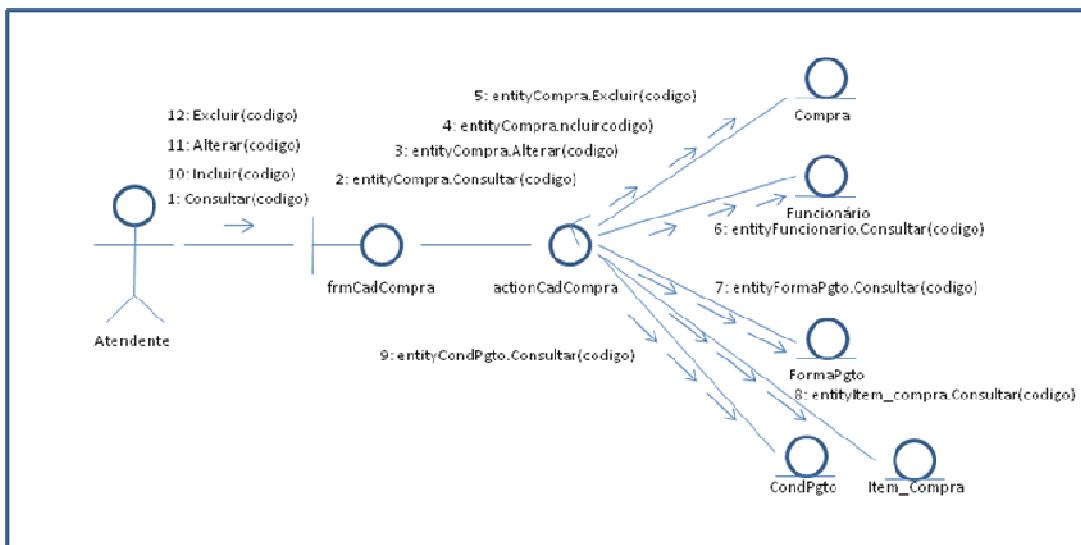


Figura 37: Diagrama de Colaboração – Manter cadastro de nota fiscal de entrada.

Fonte: Guilherme Post

Manter cadastro de produtos.

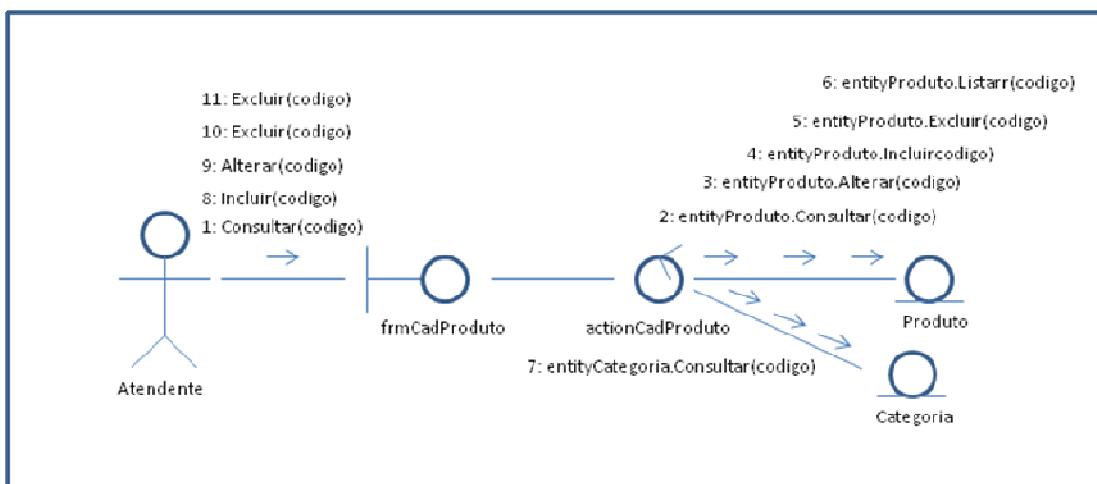


Figura 38: Diagrama de Colaboração – Manter controle de produtos.

Fonte: Guilherme Post

Manter controle de estoque.

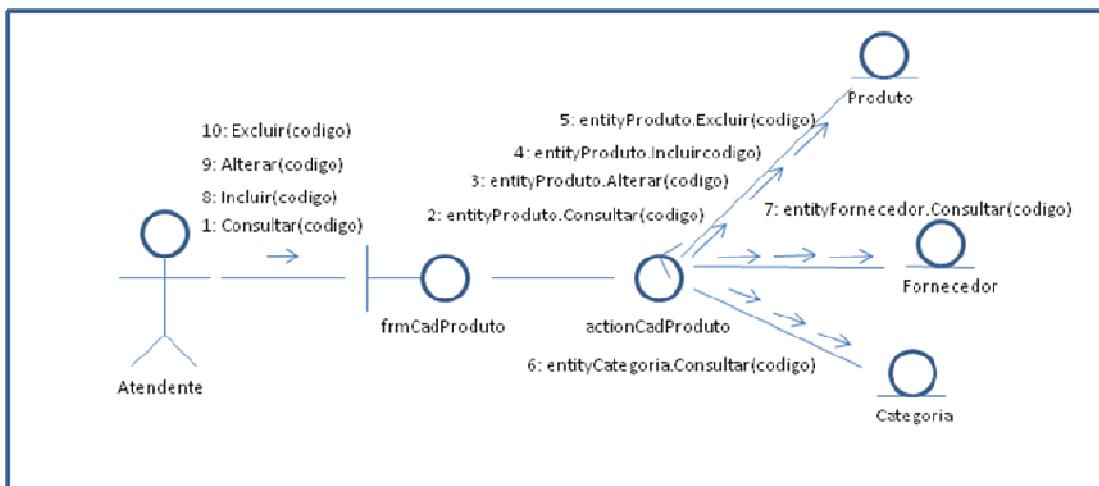


Figura 39: Diagrama de Colaboração – Manter controle estoque.
Fonte: Guilherme Post

Consultar compras.

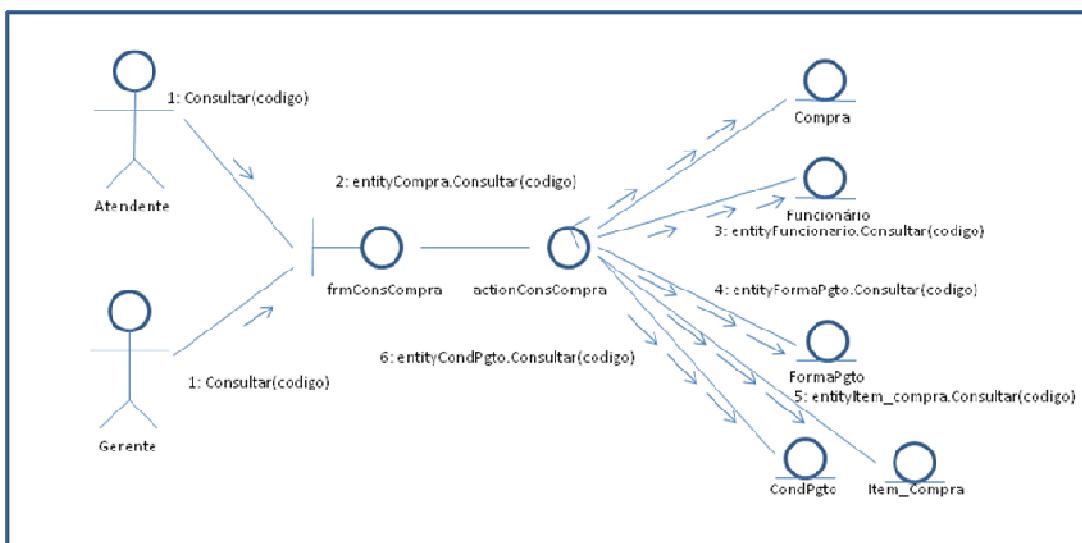


Figura 40: Diagrama de Colaboração – Consultar compras.
Fonte: Guilherme Post

Consultar clientes

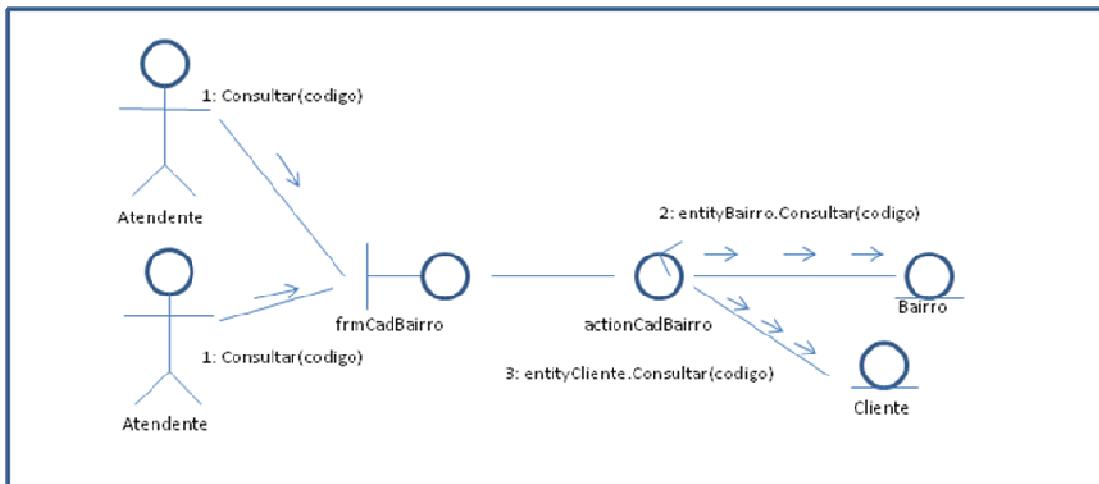


Figura 41: Diagrama de Colaboração – Consultar clientes.
Fonte: Guilherme Post

Consultar Fluxo de Caixa.

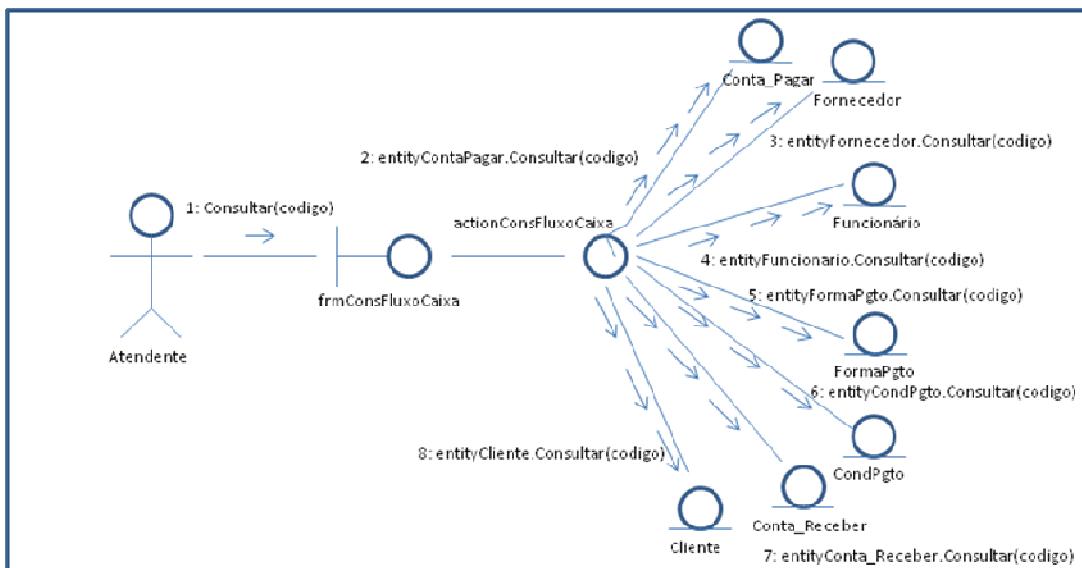


Figura 42: Diagrama de Colaboração – Consultar fluxo de caixa.
Fonte: Guilherme Post

Consultar Vendas.

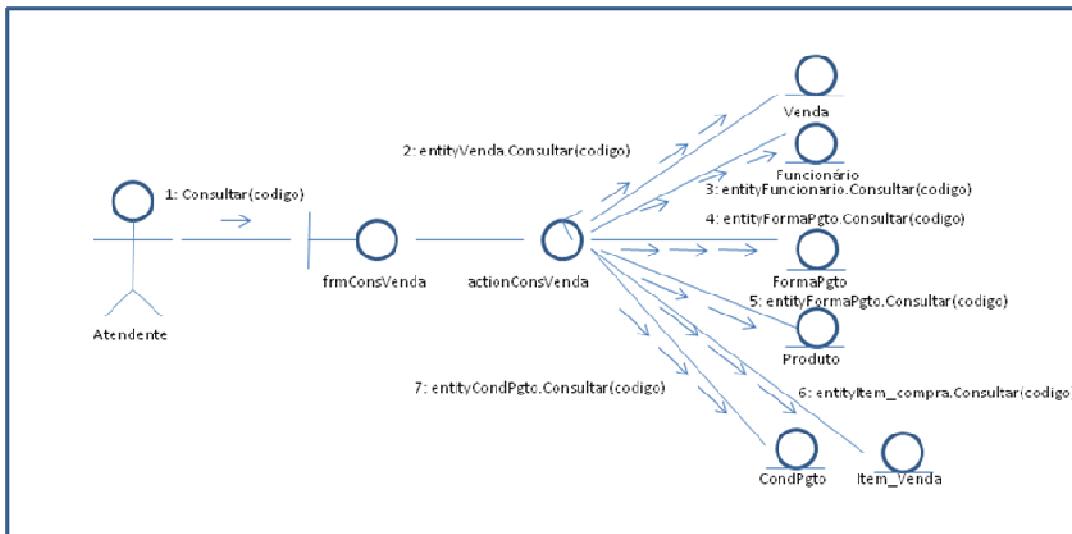


Figura 43: Diagrama de Colaboração – Consultar vendas.

Fonte: Guilherme Post

Consultar Produto.

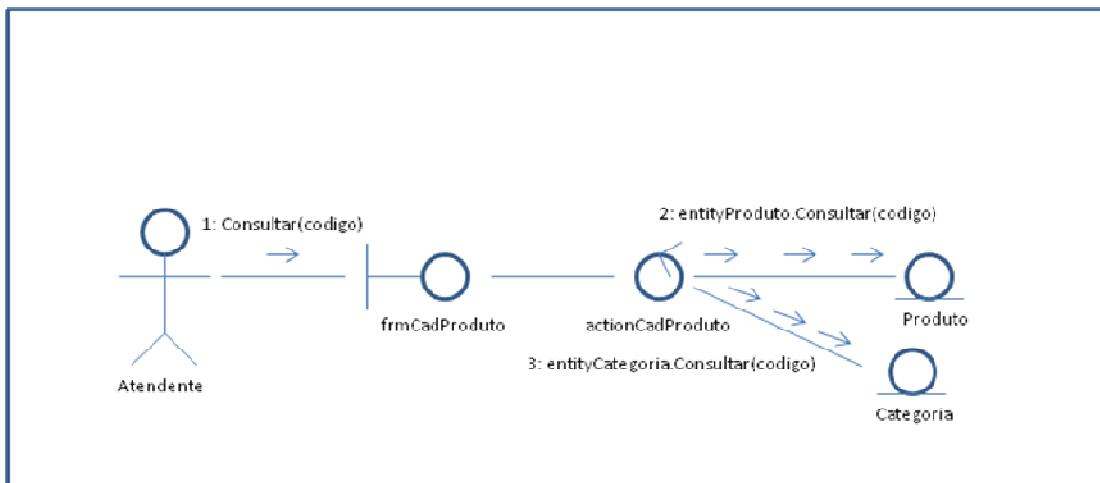


Figura 44: Diagrama de Colaboração – Consultar vendas.

Fonte: Guilherme Post

Consulta de Contas a pagar.

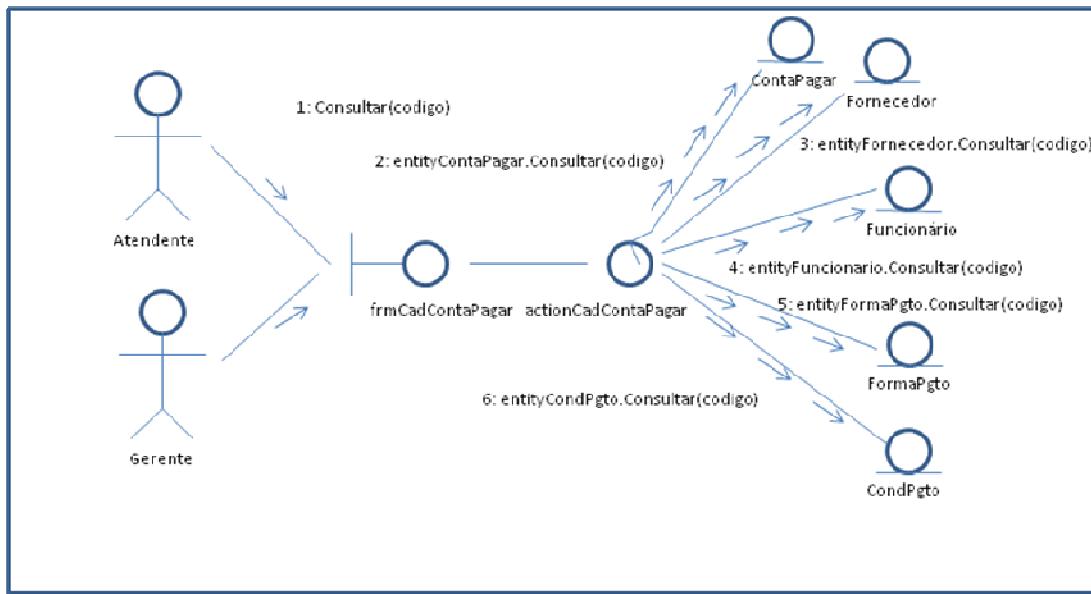


Figura 45: Diagrama de Colaboração – Consultar de contas a pagar.

Fonte: Guilherme Post

Consulta de contas a receber.

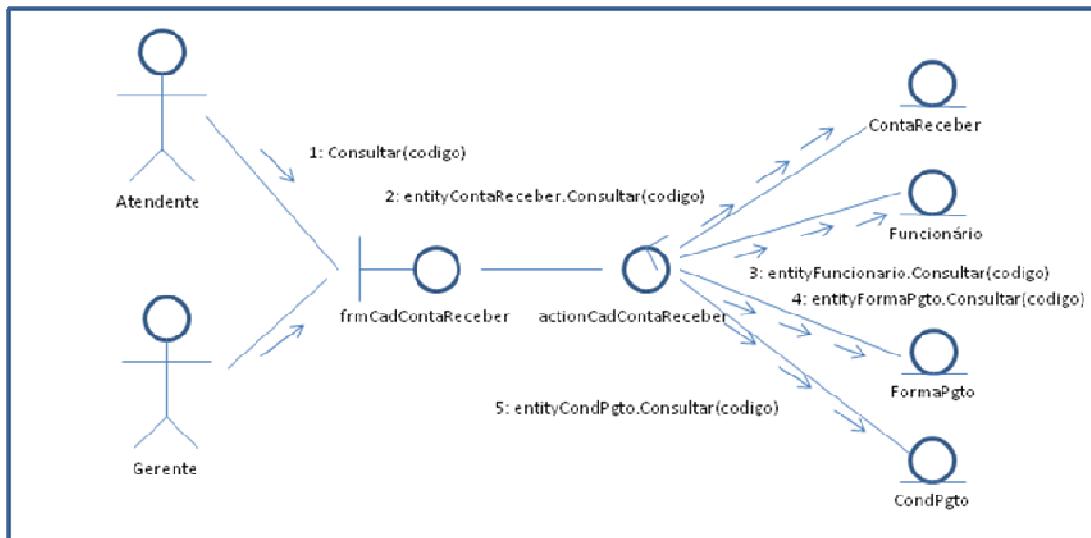


Figura 46: Diagrama de Colaboração – Consultar de contas a receber.

Fonte: Guilherme Post

Manter controle de vendas com cheque.

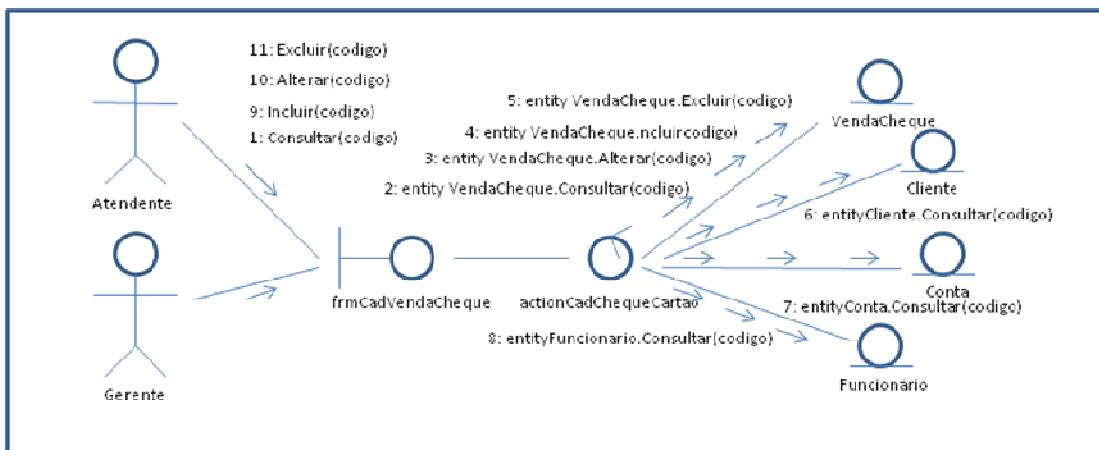


Figura 47: Diagrama de Colaboração – Manter cadastro de vendas feitas com cheque.

Fonte: Guilherme Post

Gerar relatório de contas a receber.

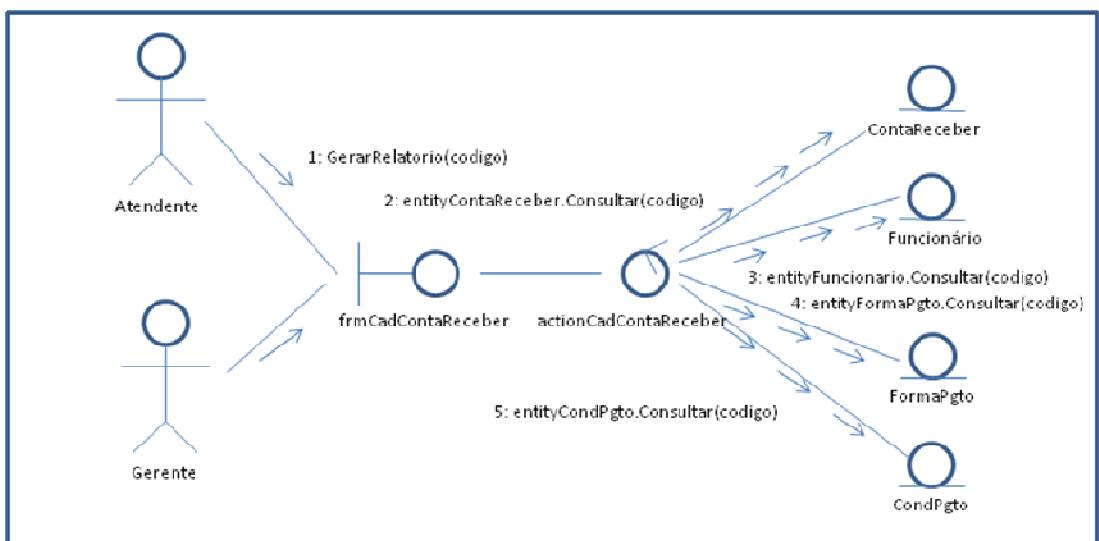


Figura 48: Diagrama de Colaboração – Gerar relatório de contas a receber.

Fonte: Guilherme Post

Gerar Relatório de contas a pagar.

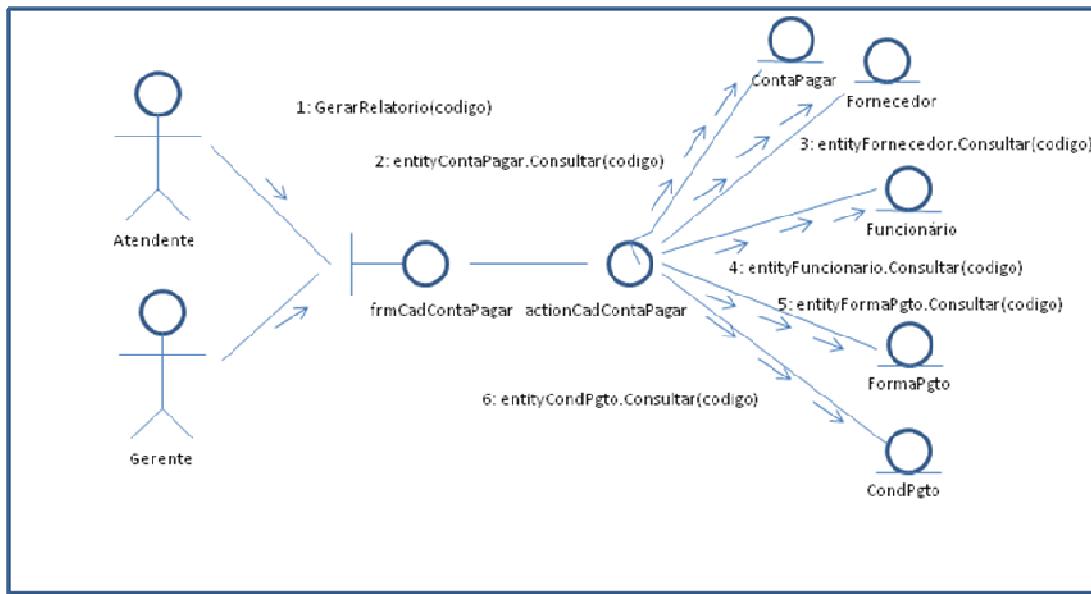


Figura 49: Diagrama de Colaboração – Gerar relatório de contas a pagar.

Fonte: Guilherme Post

Gerar relatório de fluxo de caixa.

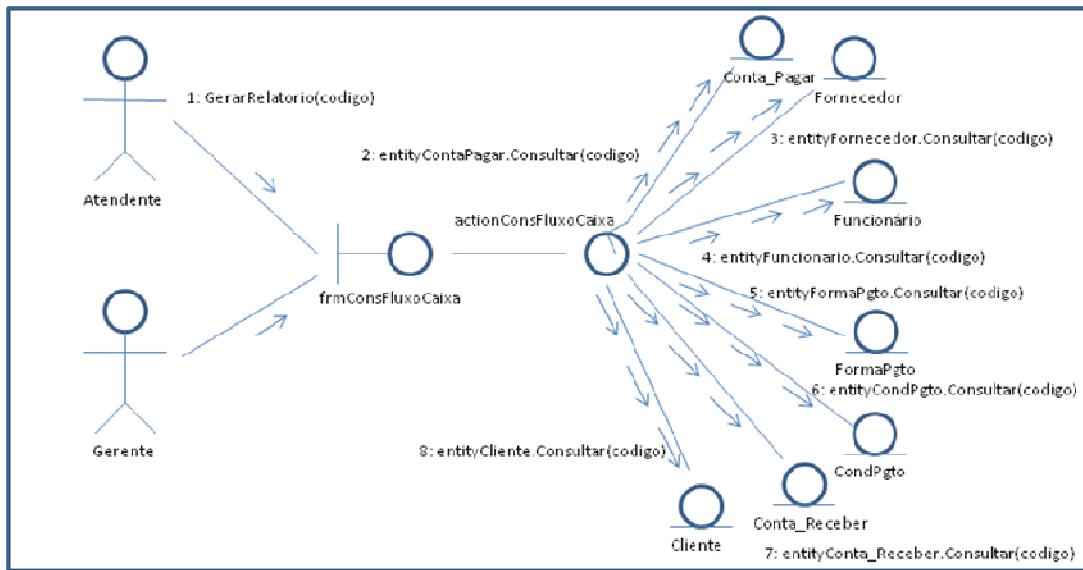


Figura 50: Diagrama de Colaboração – Gerar relatório de fluxo de caixa.

Fonte: Guilherme Post

Gerar relatório de itens em falta no estoque.

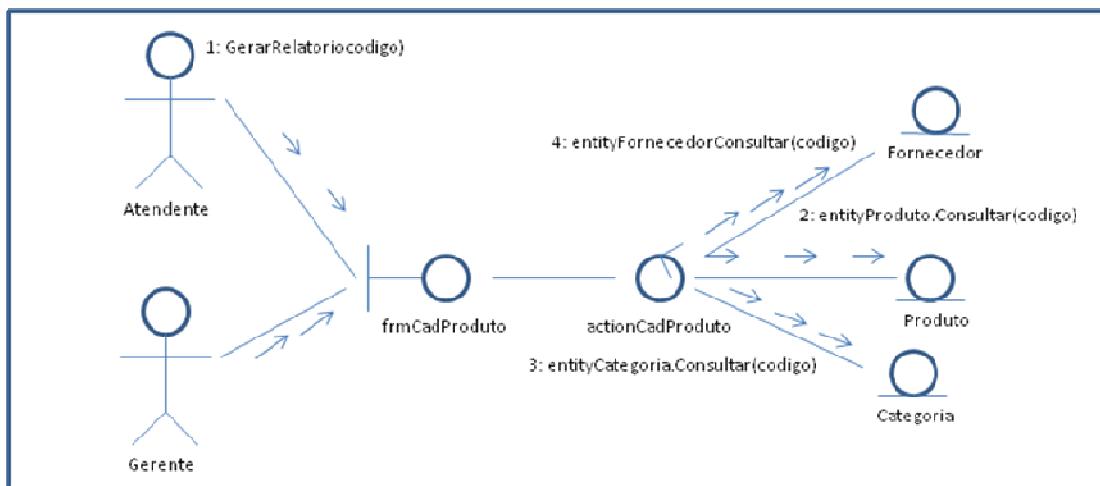


Figura 51: Diagrama de Colaboração – Gerar relatório de itens em falta no estoque.
Fonte: Guilherme Post

3.5.5. Analisar Classe

Nesta etapa são apresentados todos os atributos, métodos que poderão ser realizados sobre as classes de domínio do escopo de negócio da Agropecuária na fase de levantamento de requisitos. O diagrama de classe é de muita importância para fase de implementação, se este diagrama estiver bem definido fica mais fácil para o programador entender a interação entre as classes que compõem o sistema.

Na figura 51 são apresentadas as classes detalhadamente assim como no modelo de domínio, mas agora mostrando os métodos.

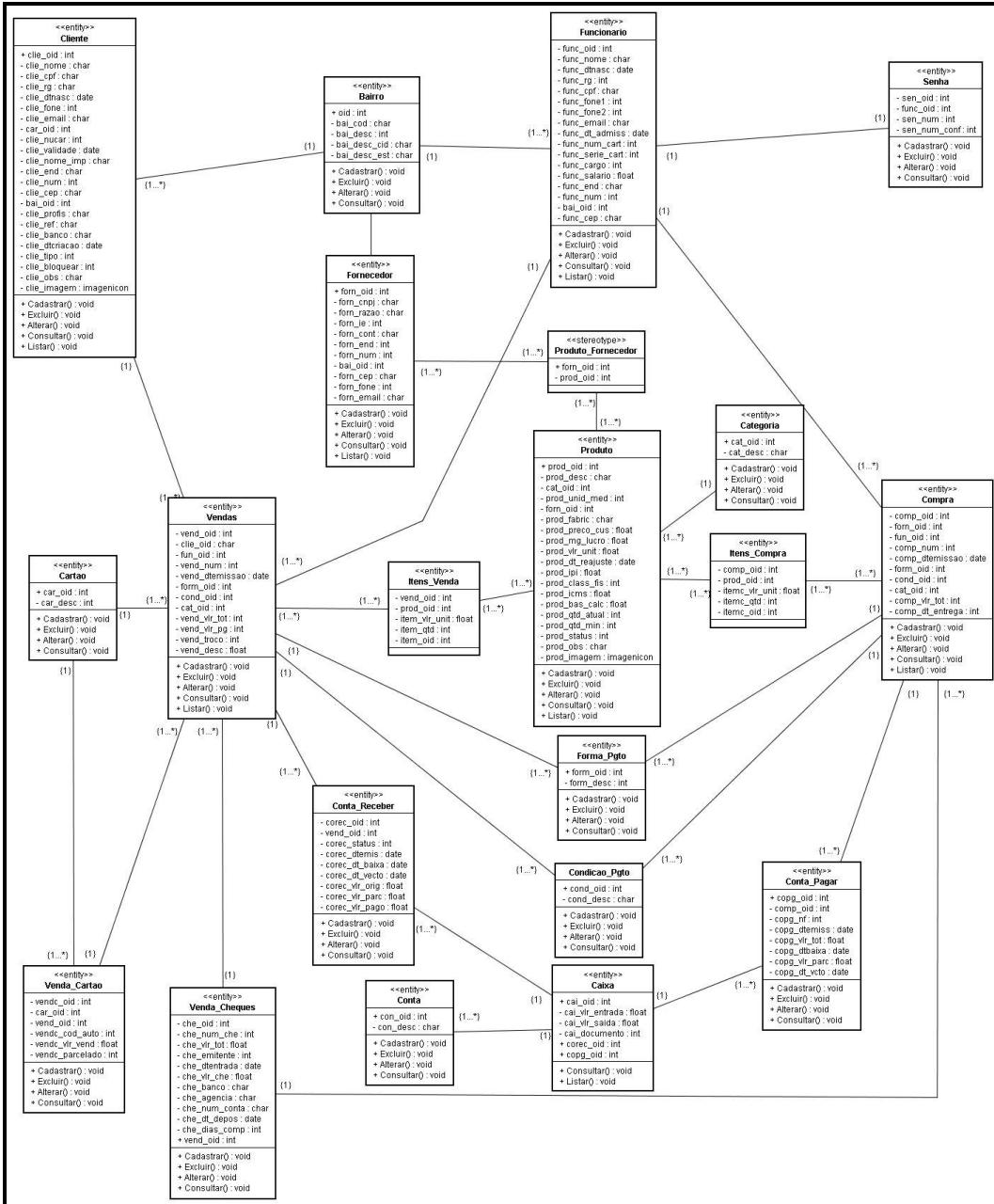


Figura 52: Diagrama de classe detalhado
Fonte: Guilherme Post

3.6. Projeto (*Design*)

É nesta etapa que são trabalhadas todas as informações obtidas no levantamento de requisitos e na análise, onde serão gerados documentos que definiram os procedimentos e métodos a serem empregados para o desenvolvimento do software, buscando deixar toda a documentação de uma forma bem clara e de fácil percepção do que é realmente necessário se

desenvolver. No projeto de banco de dados é desenvolvido o modelo entidade relacionamento do sistema e no projeto visual são mostradas as telas de interação com o usuário.

3.6.1. Projetar Banco de Dados

O banco de dados escolhido para armazenar as informações inseridas pelos usuários foi o SQL Server, seu Sistema de Gerenciamento de Banco de Dados possui uma interface e linguagem de fácil entendimento.

Nas etapas a seguir é apresentado o modelo relacional do banco de dados na terceira forma normal, MER (modelo entidade relacionamento), dicionário de dados que detalha a descrição dos atributos que compõem as classes do sistema.

3.6.2. Modelagem de dados (Normalização – 3FN)

A modelagem de dados tem o objetivo de normalizar um conjunto de dados, fazendo com que o conjunto de dados não tenham dependência entre itens de dados a não ser com a chave primária, sendo assim o que se chama de terceira forma normal.

A modelagem de dados é apresentada na figura 52. Os campos são apresentados de tal forma que, a descrição dos atributos foi formada da seguinte forma, as primeiras letras é a identificação da tabela a qual pertence o atributo e as letras na seqüência identifica o que o campo armazenará de informação. Exemplo: o campo do nome do cliente é apresentado assim clie + nome = clie_nome, sendo que o campo clie significa a tabela cliente e nome se refere a informação que será armazenada no campo, no caso nome do cliente.

Cliente	Produto	Funcionario	Fornecedor	Venda
clie_oid	prod_oid	func_oid	forn_oid	vend_oid
clie_nome	prod_desc	func_nome	forn_cnpj	clie_oid
clie_cpf	cat_oid	func_dtnasc	forn_razao	func_oid
clie_rg	prod_unid_med	func_rg	forn_ie	vend_num
clie_dtnasc	forn_oid	func_cpf	forn_cont	vend_dtemissao
clie_fone	prod_fabric	func_fone1	forn_end	form_oid
clie_email	prod_preco_cus	func_fone2	forn_num	cond_oid
car_oid	prod_mg_lucro	func_email	bai_oid	vend_vlr_tor
clie_nucar	prod_vlr_unit	func_dt_emiss	forn_cep	vendvlr_pg
clie_validade	prod_dt_reajuste	func_num_cart	forn_fone	vend_desc
clie_nome_imp	prod_ipi	func_serie_cart	forn_email	
clie_end	prod_class_fis	func_cargo		
clie_num	prod_icms	func_salario		
clie_cep	prod_bas_calc	func_end		
bai_oid	prod_qtd_atual	func_num		
clie_profits	prod_qtd_min	bai_oid		
clie_ref	prod_status	func_cep		
clie_banco	prod_obs			
clie_dtcriacao	prod_imagem			
clie_tipo				
clie_bloquear				
clie_obs				
clie_imagem				
Venda_Cheque	Conta_Receber	Compra	Conta_Pagar	Venda_Cartao
che_oid	corec_oid	comp_oid	copg_oid	vendc_oid
che_num_che	vend_oid	forn_oid	comp_oid	car_oid
che_vlr_tot	corec_status	func_oid	copg_status	vendc_cod_auto
clie_oid	corec_dtemis	comp_num	copg_nf	vendc_vlr_vend
che_emitente	corec_dt_baixa	comp_dtemis	copg_vlr_tot	vendc_parcelado
che_dtentrada	corec_dt_vecto	form_oid	copg_vlr_par	vendc_num_parc
che_vlr_che	corec_vlr_orig	cond_oid	copg_dt_emis	
che_banco	corec_vlr_parc	comp_vlr_tot	copg_dt_vcto	
che_agencia	corec_vlr_pago	comp_dt_ent		
che_num_conta				
che_dt_depos				
che_dias_comp				
com_oid				
Categoria	Forma_Pgto	Cond_Pgto	Cartao	Conta
cat_oid	form_oid	cond_oid	car_oid	con_oid
cat_desc	form_desc	cond_desc	car_desc	con_desc
Caixa	Item_Venda	Item_Compra	Senha	Prod_Fornec
cai_oid	Item_oid	itemc_oid	sen_oid	prfo_oid
cai_vlr_entrad	vend_oid	comp_oid	func_oid	prod_oid
cai_vlr_saida	prod_oid	prod_oid	sen_num	forn_oid
cai_documento	item_vlr_unit	itemc_vlr_unit	sen_num_conf	
corec_oid	item_qtd	itemc_qtd		
copg_oid				

Figura 53: Modelagem de dados na 3FN

Fonte: Guilherme Post

3.6.3. MER (Modelo Entidade Relacionamento)

O MER (Modelo Entidade Relacionamento) apresenta o relacionamento entre as entidades, com este modelo é apresentado às entidades de uma forma mais limpa e legível possibilitando um maior entendimento da interação entre entidades, é visto na figura 53.

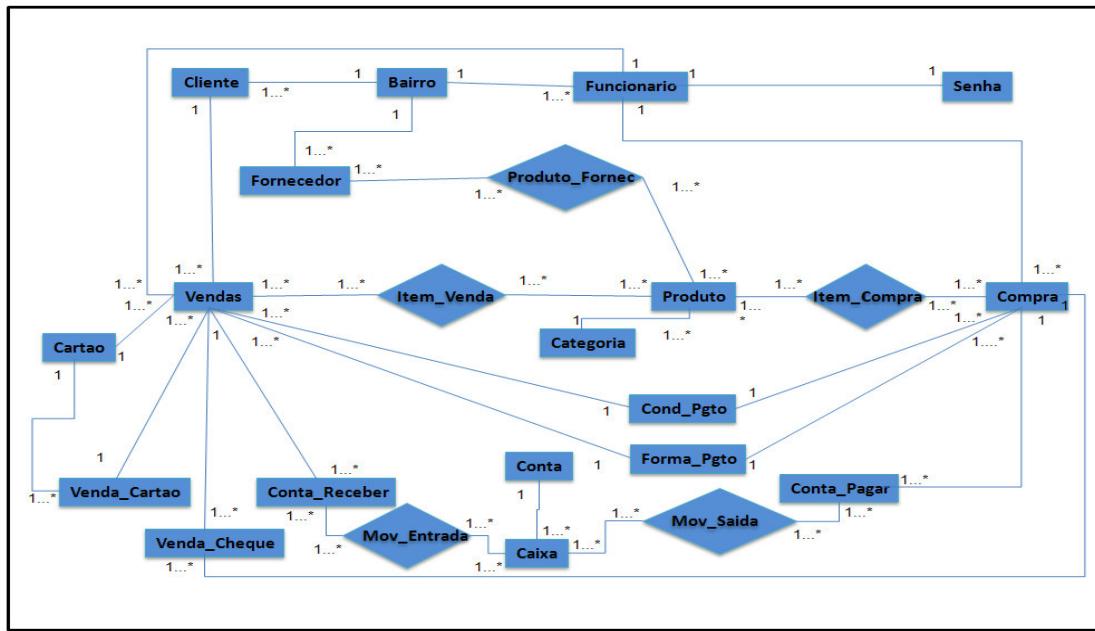


Figura 54: Modelo Entidade Relacionamento

Fonte: Guilherme Post

3.6.4. Dicionário de dados

O dicionário de dados detalhada os atributos apresentados nas classes, apresentando o nome da classe, o nome do atributo, a descrição do atributo, seu formato que pode ser numérico (representado por 9999) ou alfanumérico (representado X(50) e observações que informa se o campo é obrigatório ou não, quando é chave estrangeira ou primária. O dicionário de dados é apresentado nas tabelas a seguir.

Tabela que armazena os dados dos clientes, Cliente.

Tabela 99: Dicionário de dados da tabela cliente

Tabela: Cliente

Descrição: Armazena os dados referente aos clientes.

Atributo	Descrição	Formato	Observação
clie_oid	Código do cliente	999999999	Campo obrigatório, chave primária
clie_nome	Nome do cliente	X(50)	Campo obrigatório.
clie_cpf	CPF do cliente	999999999-99	
clie_rg	RG do cliente	999999999	
clie_dtnasc	Data de nascimento	99/99/9999	
clie_fone	Telefone de contato	X(30)	Campo obrigatório.
clie_email	Endereço de email	X(50)	
car_oid	Código do cartão	99999999999	Chave estrangeira
clie_nucar	Número do cartão	99999999999	
clie_validade	Validade do cartão	99/99/9999	
clie_nome_imp	Nome impresso no cartão	X(50)	
clie_end	Endereço da residência (Rua)	X(50)	Campo obrigatório.
clie_num	Número da casa	9999	Campo obrigatório.
clie_cep	Número do CEP	99999999-99	
bai_oid	Código do bairro	99999	Campo obrigatório, chave primária
clie_profits	Profissão do cliente	X(50)	
clie_ref	Referências comerciais	X(50)	
clie_banco	Nome do banco que possui conta	X(50)	
clie_dtcriacao	Data do cadastro	99/99/9999	
clie_tipo	Pessoa Física ou Jurídica	9	Campo obrigatório, contendo as opções: pessoa física ou jurídica
clie bloquear	Bloquear atendimento	9	
clie_obs	Informações adicionais	X(200)	
clie_imagem	Foto do cliente	2 MB. PIXEL	

Fonte: Guilherme Post

Tabela que armazena os dados das vendas com cheques, Venda_Cheque.

Tabela 100: Dicionário de dados da tabela de cheques

Tabela: Cheque

Descrição: Armazena os dados referente aos cheques emitidos nas vendas.

Atributo	Descrição	Formato	Observação
che_oid	Código do cheque	99999999	Campo obrigatório, chave primária
che_num_che	Número de cheques emitidos p/ uma venda	99	Campo obrigatório
che_vlr_tot	Valor total do cheque	99.999,99	Campo obrigatório
che_emitente	Emitente do cheque	X(50)	Campo obrigatório
che_dtentrada	Data de entrada do cheque	99/99/9999	Campo obrigatório
che_vlr_che	Valor de cada cheque	99.999,99	Campo obrigatório
che_banco	Banco a qual pertence o cheque	X(50)	Campo obrigatório
che_agencia	Agência bancária	999999999	Campo obrigatório
che_num_conta	Número da conta bancária	999999999	Campo obrigatório
che_dt_depos	Data a ser depositado	99/99/9999	Campo obrigatório
che_dias_comp	Dias para compensação	99	Campo obrigatório
vend_oid	Código da venda	999999999	Campo obrigatório, chave estrangeira

Fonte: Guilherme Post

Tabela que armazena os dados do fluxo de caixa, Caixa.

Tabela 101: Dicionário de dados da tabela de caixa

Tabela: Caixa

Descrição: Armazena os dados referente fluxo de caixa.

Atributo	Descrição	Formato	Observação
cai_oid	Código da movimentação do caixa	99999999	Campo obrigatório, chave primária
cai_vlr_entraida	Valor da entrada do caixa	99.999,99	Campo obrigatório
cai_vlr_saida	Valor de saída do caixa	99.999,99	Campo obrigatório
cai_documento	Documento	X(50)	Campo obrigatório
corec_oid	Código da conta a receber	999999999	Campo obrigatório, chave estrangeira
copg_oid	Código da conta a pagar	999999999	Campo obrigatório, chave estrangeira

Fonte: Guilherme Post

Tabela que armazena os dados dos produtos, Produto.

Tabela 102: Dicionário de dados da tabela de produtos

Tabela: Produto

Descrição: Armazena os dados referente aos produtos que possui na empresa.

Atributo	Descrição	Formato	Observação
prod_oid	Código do produto	999999999	Campo obrigatório, chave primária
prod_desc	Descrição do produto	X(50)	Campo obrigatório
cat_oid	Categoria a qual pertence o produto	999999999	Campo obrigatório, chave estrangeira
prod_unid_med	unidade de medida do produto	9	Campo obrigatório
forn_oid	Código do fornecedor	999999999	Chave estrangeira
prod_fabric	Fabricante do produto	X(50)	
prod_preco_cus	Preço de custo	99.999,99	Campo obrigatório
prod_mg_lucro	Margem de lucro utilizada no produto	99,99	Campo obrigatório
prod_vlr_unit	Valor unitário a ser vendido	99.999,99	Campo obrigatório
prod_dt_reajuste	Data do ultimo reajuste do produto	99/99/9999	Campo obrigatório
prod_ipi	Percentual do IPI	99,99	
prod_class_fis	Classificação fiscal do produto	9999999	
prod_icms	Percentual do ICMS	99,99	
prod_bas_calc	Base de calculo do produto	999.999,99	
prod_qtd_atual	Quantidade atual no estoque	9999999999	Campo obrigatório
prod_qtd_min	Quantidade mínima no estoque	9999999999	Campo obrigatório
prod_status	Produto está ativo ou inativo	9	Campo obrigatório
prod_obs	Observações do produto	X(250)	
prod_imagem	Foto do produto	2 MB. PIXEL	

Fonte: Guilherme Post

Tabela que armazena os dados dos cartões, Cartão.

Tabela 103: Dicionário de dados da tabela de cartão

Tabela: Cartão

Descrição: Armazena os dados referente aos cartões que a empresa oferece (Crédito ou Débito).

Atributo	Descrição	Formato	Observação
car_oid	Código do cartão	999999999	Campo obrigatório, chave primária
car_desc	Descrição do cartão	X(50)	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados dos funcionários, Funcionário.

Tabela 104: Dicionário de dados da tabela de funcionários

Tabela: Funcionário**Descrição:** Armazena os dados referente aos funcionários.

Atributo	Descrição	Formato	Observação
func_oid	Código do funcionário	999999999	Campo obrigatório, chave primária
func_nome	Nome	X(50)	Campo obrigatório
func_dtnasc	Data de nascimento	99/99/9999	Campo obrigatório
func_rg	RG do funcionário	999999999	Campo obrigatório
func_cpf	CPF do funcionário	999999999-99	Campo obrigatório
func_fone1	Telefone de contato	X(30)	Campo obrigatório
func_fone2	Celular	X(30)	Campo obrigatório
func_email	Endereço de e-mail	X(50)	Campo obrigatório
func_dt_admiss	Data de admissão	99/99/9999	Campo obrigatório
func_num_cart	Numero da carteira de trabalho	99999999	Campo obrigatório
func_serie_cart	Série da carteira de trabalho	999999	Campo obrigatório
func_cargo	Cargo do funcionário	X(30)	Campo obrigatório
func_salario	Valor do Salário	99.999,99	Campo obrigatório
func_end	Endereço da residência (Rua)	X(50)	Campo obrigatório
func_num	Número da residência	9999	Campo obrigatório
bai_oid	Código do bairro	999999999	Campo obrigatório, chave estrangeira
func_cep	Número do CEP	99999999-99	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados da forma de pagamento, Forma_Pgto.

Tabela 105: Dicionário de dados da tabela de forma de pagamento

Tabela: Forma_Pgto**Descrição:** Armazena os dados referente a forma de pagamento.

Atributo	Descrição	Formato	Observação
form_oid	Código da forma de pagamento	999999999	Campo obrigatório, chave primária
form_desc	Descrição da forma de pagamento	X(50)	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados dos fornecedores, Fornecedor.

Tabela 106: Dicionário de dados da tabela de fornecedores

Tabela: Fornecedor

Descrição: Armazena os dados referente aos fornecedores.

Atributo	Descrição	Formato	Observação
forn_oid	Código do fornecedor	999999999	Campo obrigatório, chave primária
forn_cnpj	CNPJ do fornecedor	99.999.999/9999-99	Campo obrigatório
forn_razao	Razão Social do fornecedor	X(50)	Campo obrigatório
forn_ie	Inscrição Estadual do fornecedor	99999999999	
forn_cont	Pessoa de contato (representante)	X(50)	Campo obrigatório
forn_end	Endereço da empresa (rua)	X(50)	
forn_num	Número do endereço	9999	
bai_oid	Código do bairro	999999999	Campo obrigatório, chave estrangeira
forn_cep	Número do CEP	99999999-99	
forn_fone	Telefone da empresa e contato	X(50)	Campo obrigatório
forn_email	Endereço de e-mail da empresa ou contato	X(100)	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados das contas a pagar, Conta_Pagar.

Tabela 107: Dicionário de dados da tabela de contas a pagar

Tabela: Conta_Pagar

Descrição: Armazena os dados referente as contas a pagar da empresa.

Atributo	Descrição	Formato	Observação
copg_oid	Código da conta a pagar	999999999	Campo obrigatório, chave primária
comp_oid	Código da compra	999999999	Campo obrigatório, chave estrangeira
copg_nf	Número da nota fiscal de entrada	999999999	Campo obrigatório
copg_vlr_tot	Valor total da conta a pagar	99.999,99	Campo obrigatório
copg_vlr_par	Valor da parcela da conta a pagar	99.999,99	Campo obrigatório
copg_dt_emis	Data da emissão da nota fiscal	99/99/9999	Campo obrigatório
copg_dt_vcto	Data do vencimento	99/99/9999	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados das vendas, Venda.

Tabela 108: Dicionário de dados da tabela de vendas

Tabela: Venda			
Descrição: Armazena os dados referente as vendas.			
Atributo	Descrição	Formato	Observação
vend_oid	Código da venda	999999999	Campo obrigatório, chave primária
clie_oid	Código do cliente	999999999	Campo obrigatório, chave estrangeira
func_oid	Código do funcionário	999999999	Campo obrigatório, chave estrangeira
vend_num	Número da venda	999999999	Campo obrigatório
vend_dtemissao	Data de emissão da venda	99/99/9999	Campo obrigatório
form_oid	Código da forma de pagamento	999999999	Campo obrigatório, chave estrangeira
cond_oid	Código da condição de pagamento	999999999	Campo obrigatório, chave estrangeira
vend_vlr_tor	Valor total da venda	99.999,99	Campo obrigatório, chave estrangeira
vendvlr_pg	Valor pago pelo cliente	99.999,99	Campo obrigatório
vend_desc	Percentual do desconto	99,99	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados dos itens das vendas, Item_Venda.

Tabela 109: Dicionário de dados da tabela de item da venda

Tabela: Item_Venda			
Descrição: Armazena os dados referente aos itens da venda.			
Atributo	Descrição	Formato	Observação
Item_oid	Código do item da venda	999999999	Campo obrigatório, chave primária
vend_oid	Código da venda	999999999	Campo obrigatório, chave estrangeira
prod_oid	Código do produto	999999999	Campo obrigatório, chave estrangeira
item_vlr_unit	Valor unitário do produto	99.999,99	Campo obrigatório
item_qtd	Quantidade vendida do produto	999999999	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados das contas a receber, Conta_Receber.

Tabela 110: Dicionário de dados da tabela de conta a receber

Tabela: Conta_Receber

Descrição: Armazena os dados referente as contas a receber geradas pelas vendas.

Atributo	Descrição	Formato	Observação
corec_oid	Código da conta a receber	999999999	Campo obrigatório, chave primária
vend_oid	Código da venda	999999999	Campo obrigatório, chave estrangeira
corec_status	Apresenta status da conta a receber	X(10)	Campo obrigatório, apresentando as opções: aberta ou fechada
corec_dtemis	Data emissão da conta a receber	99/99/9999	Campo obrigatório
corec_dt_baixa	Data na qual foi paga a conta	99/99/9999	Campo obrigatório
corec_dt_vecto	Data de vencimento da conta	99/99/9999	Campo obrigatório
corec_vlr_orig	Valor original da conta	99.999,99	Campo obrigatório
corec_vlr_parcela	Valor da parcela	99.999,99	Campo obrigatório
corec_vlr_pg	Valor pago pelo cliente	99.999,99	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados das compras, Compra.

Tabela 111: Dicionário de dados da tabela de compras

Tabela: Compra

Descrição: Armazena os dados referente as compras feitas pela empresa.

Atributo	Descrição	Formato	Observação
comp_oid	Código da compra	999999999	Campo obrigatório, chave primária
forn_oid	Código do fornecedor	999999999	Campo obrigatório, chave estrangeira
func_oid	Código do Funcionário	999999999	Campo obrigatório, chave estrangeira
comp_num	Número da compra	999999999	Campo obrigatório
comp_dtemis	Data de emissão da compra	99/99/9999	Campo obrigatório
form_oid	Código da forma de pagamento	999999999	Campo obrigatório, chave estrangeira
cond_oid	Código da condição de pagamento	999999999	Campo obrigatório
comp_vlr_tot	Valor total da compra	99.999,99	Campo obrigatório
comp_dt_ent	Data de entrega da nota fiscal	99/99/9999	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados dos itens das compras, Item_Compra.

Tabela 112: Dicionário de dados da tabela de itens da compra

Tabela: Item_Compra

Descrição: Armazena os dados referente aos itens da compra.

Atributo	Descrição	Formato	Observação
itemc_oid	Código do item da compra	999999999	Campo obrigatório, chave primária
comp_oid	Código da compra	999999999	Campo obrigatório, chave estrangeira
prod_oid	Código do produto	999999999	Campo obrigatório, chave estrangeira
itemc_vlr_unit	Valor unitário do produto comprado	99.999,99	Campo obrigatório
itemc_qtd	Quantidade do produto comprado	999999999	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados das condições de pagamento, Cond_Pgto.

Tabela: Cond_Pgto

Descrição: Armazena os dados referente as condições de pagamentos.

Atributo	Descrição	Formato	Observação
cond_oid	Código da condição de pagamento	999999999	Campo obrigatório, chave primária
cond_desc	Descrição da condição de pagamento	X(50)	Campo obrigatório

Tabela 113: Dicionário de dados da tabela de condição de pagamento

Fonte: Guilherme Post

Tabela que armazena os dados das vendas feitas com cartão, Venda_Cartao.

Tabela 114: Dicionário de dados da tabela de vendas com cartão

Tabela: Venda_Cartao

Descrição: Armazena os dados referente aos clientes.

Atributo	Descrição	Formato	Observação
vendc_oid	Código da venda	999999999	Campo obrigatório, chave primária
car_oid	Código do cartão	999999999	Campo obrigatório, chave estrangeira
vendc_num_parc	Número de parcelas	X(30)	Campo obrigatório
vendc_cod_auto	Código de autorização do uso do cartão	999999999	Campo obrigatório
vendc_vlr_vend	Valor da venda feita com cartão	99.999,99	Campo obrigatório
vendc_parcelado	Venda parcelada	9	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados das contas, Conta.

Tabela 115: Dicionário de dados da tabela de contas

Tabela: Conta

Descrição: Armazena os dados referente aos contas da empresa.

Atributo	Descrição	Formato	Observação
con_oid	Código da conta da empresa	999999999	Campo obrigatório, chave primária
con_desc	Descrição da conta	X(50)	Campo obrigatório

Fonte: Guilherme Post

Tabela que armazena os dados dos produtos e fornecedores, Produto_Fornecedor.

Tabela 116: Dicionário de dados da tabela de produtos e fornecedores

Tabela: Produto_Fornecedor

Descrição: Armazena os dados referente aos produtos fornecidos pelos fornecedores.

Atributo	Descrição	Formato	Observação
prfo_oid	Código	999999999	Campo obrigatório, chave primária
prod_oid	Código do produto	999999999	Campo obrigatório, chave estrangeira
forn_oid	Código do fornecedor	999999999	Campo obrigatório, chave estrangeira

Fonte: Guilherme Post

Tabela que armazena os dados das senhas dos funcionários, Senha.

Tabela 117: Dicionário de dados da tabela de senhas

Tabela: Senha

Descrição: Armazena os dados referente as senhas dos usuários.

Atributo	Descrição	Formato	Observação
sen_oid	Código da senha	999999999	Campo obrigatório, chave primária
func_oid	Código do funcionário	999999999	Campo obrigatório, chave estrangeira
sen_num	Primeira senha	X(6)	Campo obrigatório
sen_num_conf	Confirmação da primeira senha	X(6)	Campo obrigatório

Fonte: Guilherme Post

3.7. Projetar Interface

Nesta etapa foram desenvolvidas as telas do sistema. Como já foi analisada a necessidade do que o sistema precisa executar foi implementado as telas para apresentar aos usuários para que validem ou solicitem otimizações na tela.

Layout Menu Inicial.

O *layout* principal será o passo inicial para acessar os módulos do sistema é apresentado na figura 54. O usuário informará o nome, senha, pressionará o botão ok, caso usuário, senha e o modulo escolhido estejam corretos abrirá o modulo que selecionou.



Figura 55: *Layout Login e senha*

Fonte: Guilherme Post

Layout Principal.

A partir deste *layout* apresentado na figura 55 é possível acessar as funcionalidades do sistema, o usuário terá acesso comercial, suprimentos, recursos humanos e financeiros. Através de cada modulo poderá conforme sua permissão abrir a tela de cadastro de clientes, vendas, compras, contas a pagar, contas a receber e listar relatórios.

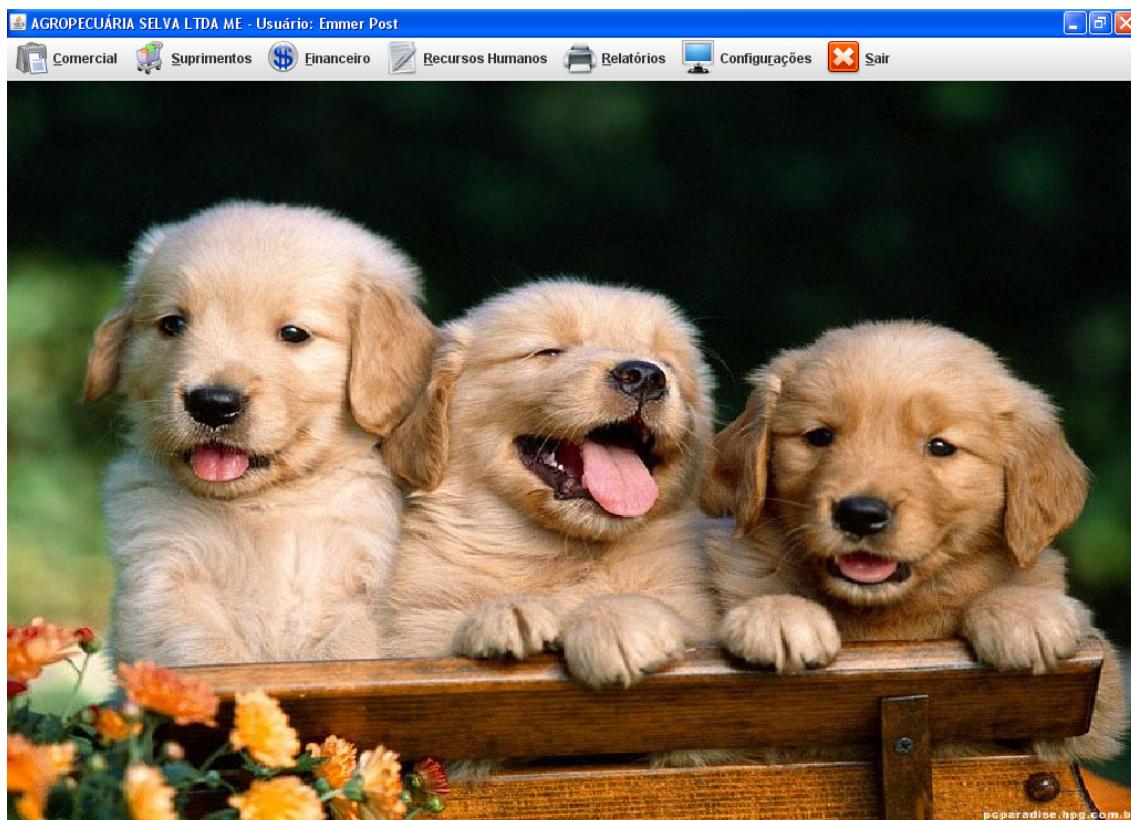


Figura 56: *Layout* principal

Fonte: Guilherme Post

Layout de cadastro de cliente.

Neste *layout* que é apresentada na figura 56 será possível o usuário a cadastrar clientes, realizar consultas, solicitar alteração do cadastro ou excluir cliente. Na parte inferior do *layout* de cliente Possui os ícones para alterar cadastro, novo que é o cadastro do novo cliente, excluir cliente, consultar cliente.

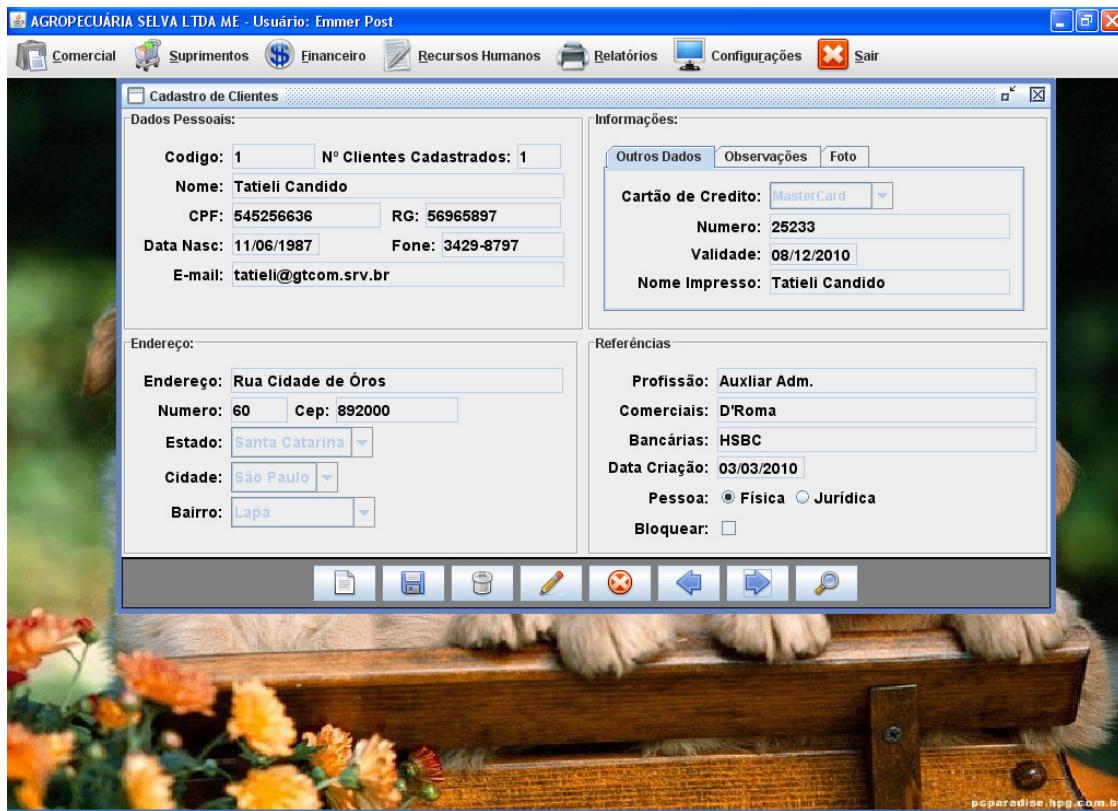


Figura 57: *Layout* cadastro de cliente

Fonte: Guilherme Post

Layout cadastro de bairros.

Neste *layout* que é apresentada na figura 57 será possível o usuário a cadastrar bairros, realizar consultas, solicitar alteração do cadastro ou excluir bairro. Na parte inferior do *layout* de bairro Possui os ícones de atalho para salvar cadastro, alterar cadastro, novo que é o cadastro do novo bairro, excluir bairro, consultar bairro.

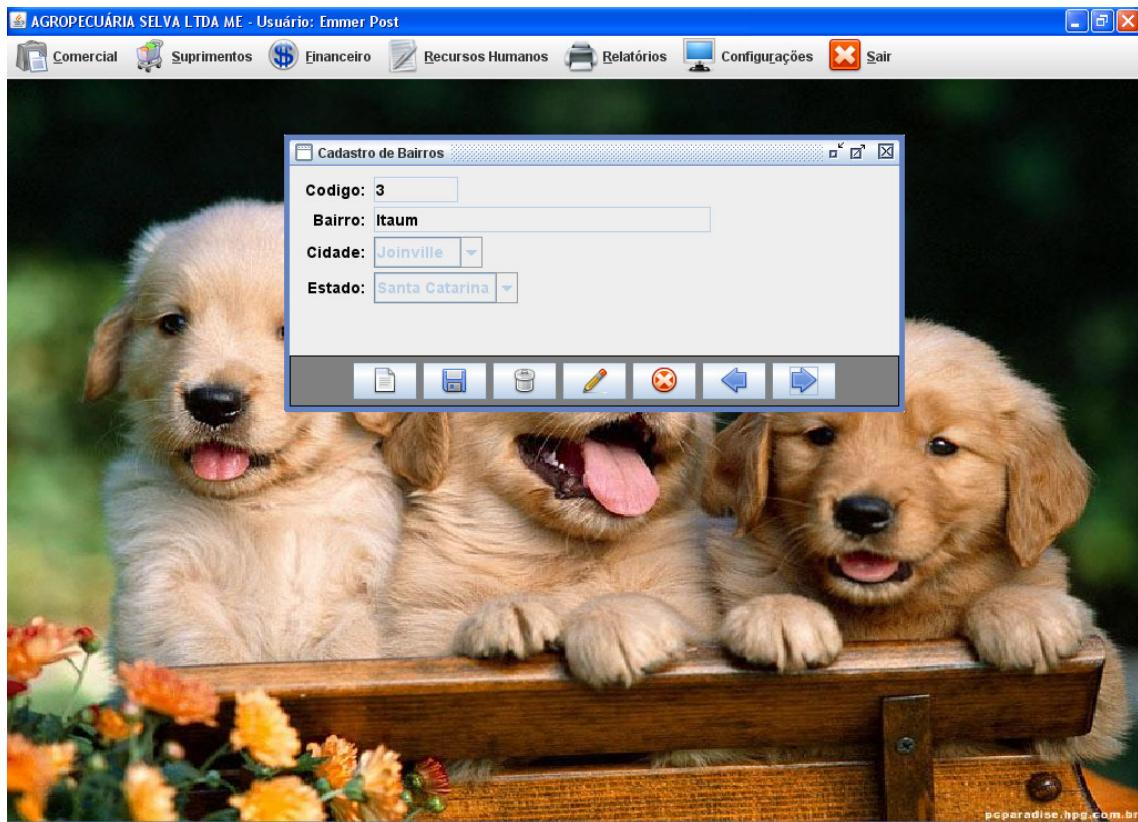


Figura 58: *Layout* cadastro de bairro
Fonte: Guilherme Post

Layout de cadastro de Vendas.

Neste *layout* que é apresentada na figura 58 será possível o usuário a cadastrar vendas, realizar consultas, solicitar alteração do cadastro ou excluir uma venda. Na parte inferior do *layout* de vendas possui os ícones de atalho para salvar cadastro, novo cadastro que é o cadastro de uma nova venda, excluir, consultar uma venda. Acima dos ícones de atalho o usuário poderá inserir ou ver os itens do pedido de venda. Nesta tela é possível visualizar o nome do cliente e seus dados, o código e nome do vendedor que emitiu a venda, os itens da venda, os valores totais e dados da venda.

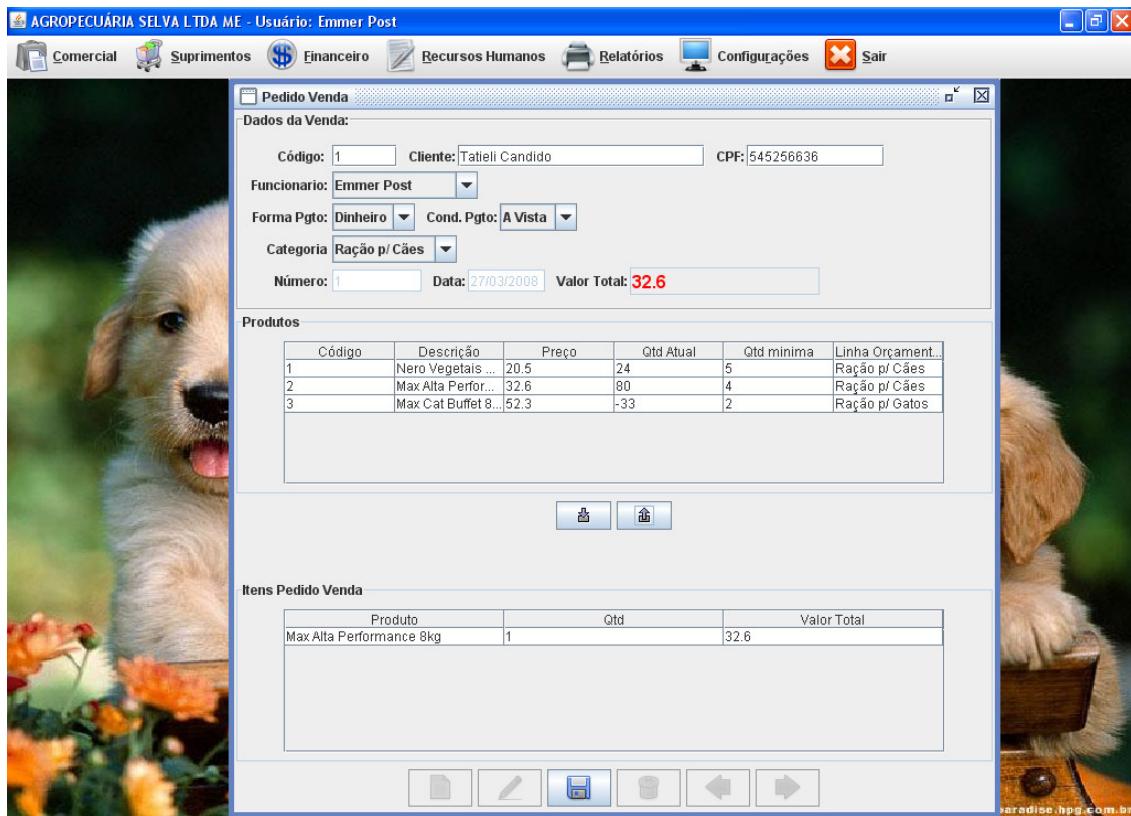


Figura 59: *Layout* cadastro de vendas

Fonte: Guilherme Post

Layout de consulta Estoque.

Neste *layout* apresentada na figura 59 será possível o usuário realizar consultas. Na parte inferior do *layout* de estoque possui os ícones de atalho para consultar um produto em estoque. Acima dos ícones de atalho o usuário visualizar o código do produto, descrição, quantidade atual e mínima no estoque e valor unitário do produto. Esta tela poderá ser solicitada pelo vendedor ao emitir a venda para certificar-se que produto em estoque, ou para o comprador solicitar um relatório dos produtos que são necessários para solicitar em uma compra para um fornecedor.

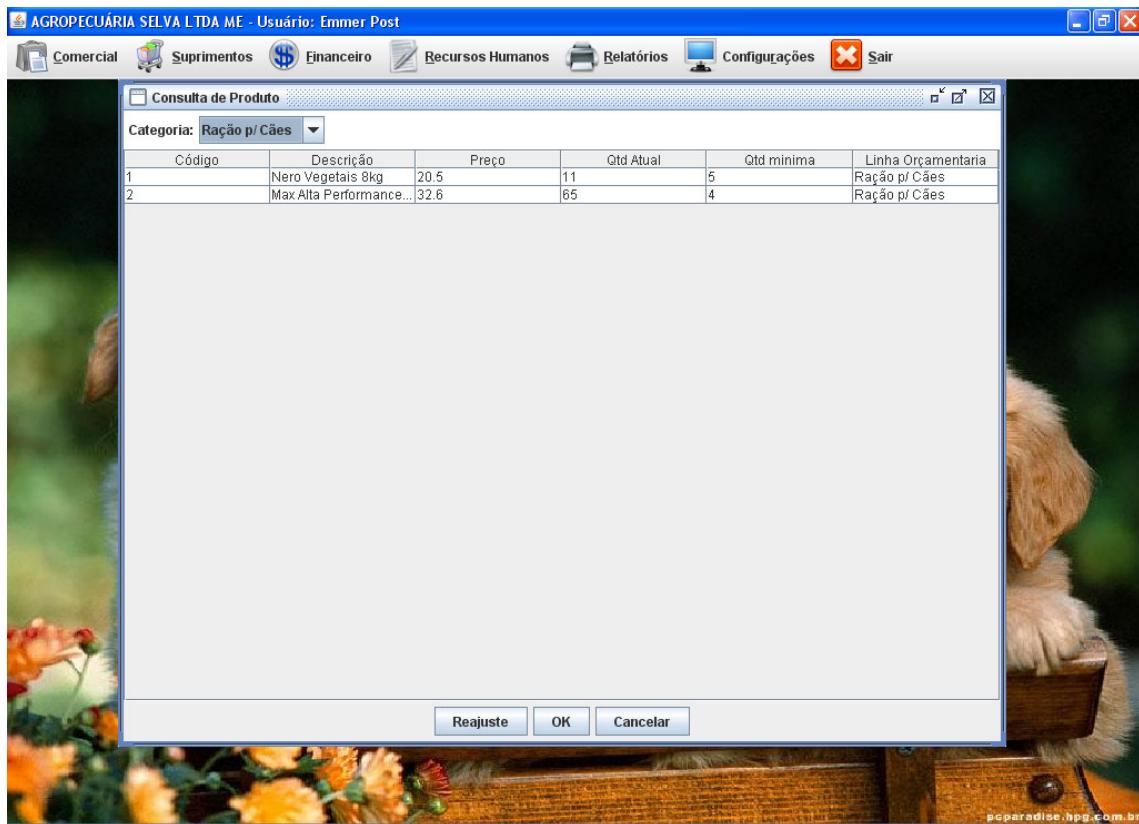


Figura 60: *Layout* controle de estoque
Fonte: Guilherme Post

Layout de cadastro de Funcionários.

Neste *layout* que é apresentada na figura60 será possível o usuário a cadastrar um funcionário, realizar consultas, solicitar alteração do cadastro ou excluir um funcionário. Na parte inferior do *layout* possui os ícones de atalho para salvar cadastro novo ou alteração, ícone para alterar cadastro, novo cadastro que é o cadastro de um novo Funcionário, excluir, consultar um funcionário. Será necessário o usuário a informar todos os dados abaixo caso contrario não poderá salvar um novo funcionário.

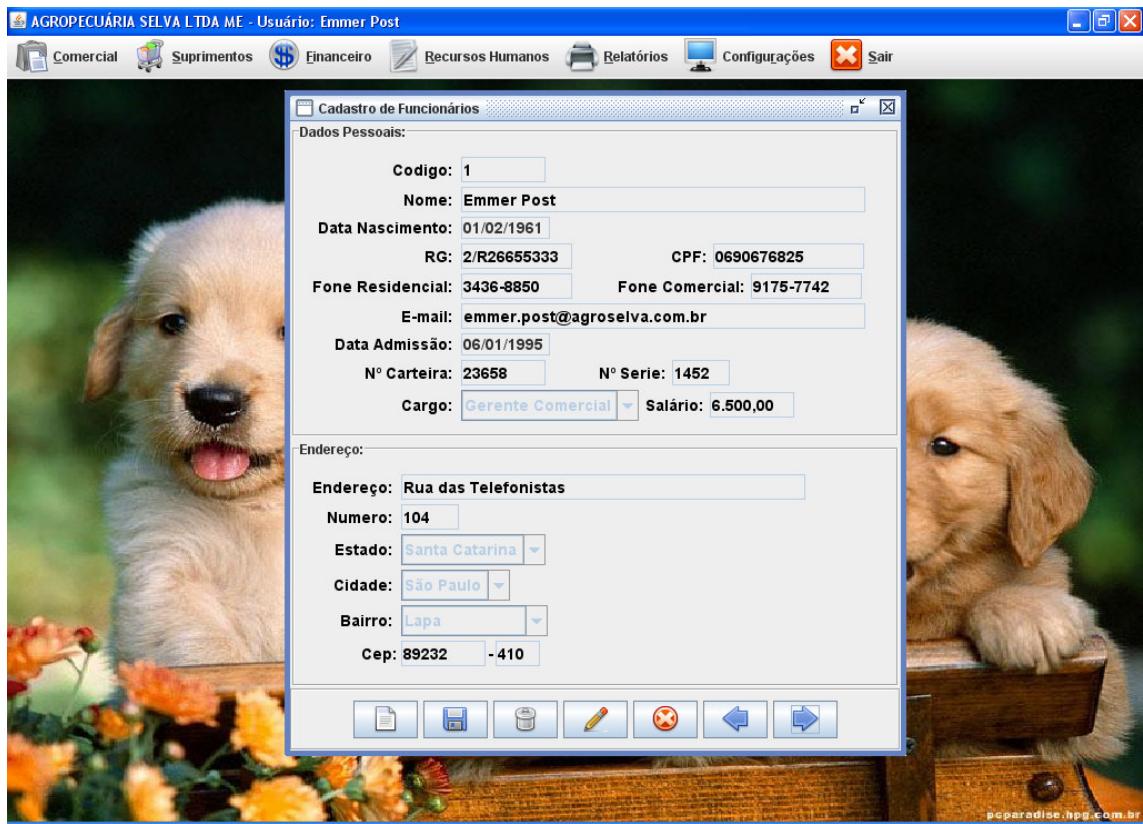


Figura 61: *Layout* cadastro de funcionário
Fonte: Guilherme Post

Layout de cadastro de senhas.

Este *layout* que é apresentado na figura 61 possibilitará o usuário a cadastrar uma senha para os funcionários que farão uso do sistema, onde informará o código do usuário, senha com no mínimo quatro e no máximo seis caracteres, selecionara o modulo que funcionários poderá acessar e o que poderá fazer ao acessar os módulos.

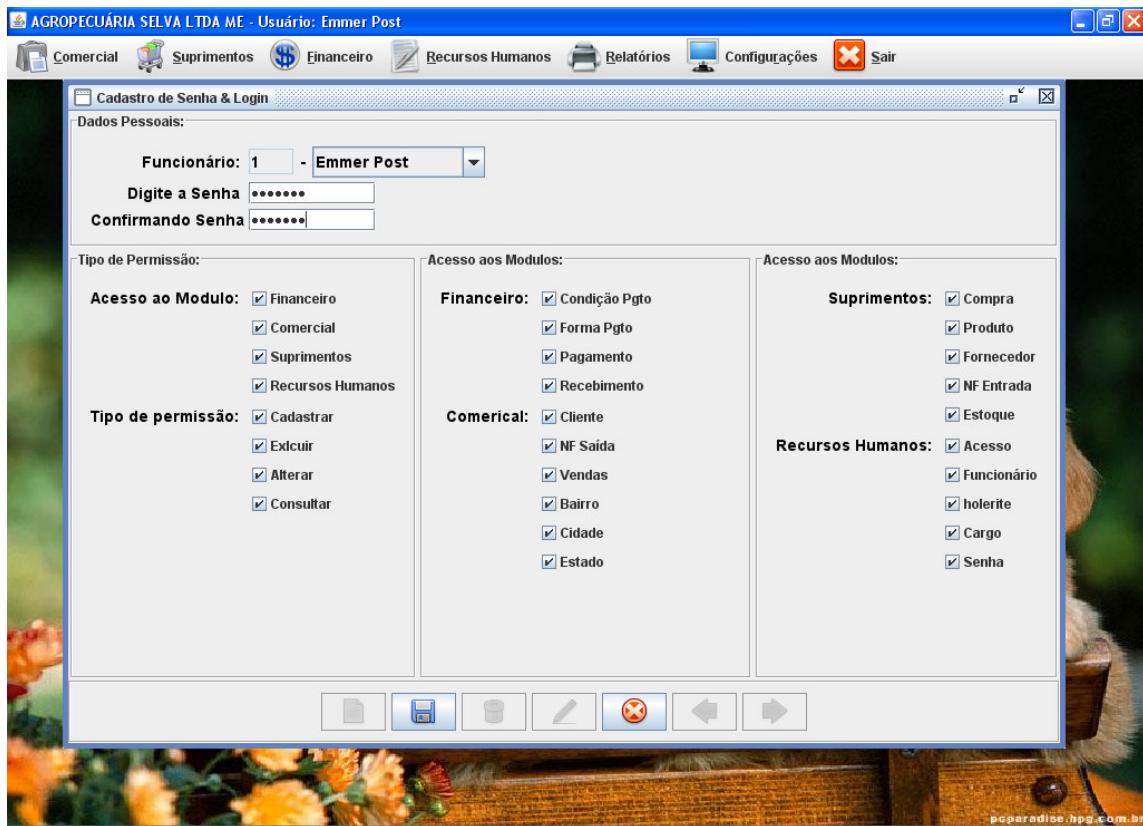


Figura 62: *Layout* cadastro de cadastro de senhas
Fonte: Guilherme Post

Layout de Contas a Pagar.

Neste *layout* que é apresentado na figura 62 será possível o usuário a cadastrar uma conta a pagar através da emissão da NF de entrada, realizar consultas, solicitar alteração do cadastro ou excluir uma conta a pagar. Na parte inferior do *layout* possui os ícones de atalho para salvar, novo cadastro que é o cadastro de uma nova conta a pagar, excluir, consulta.

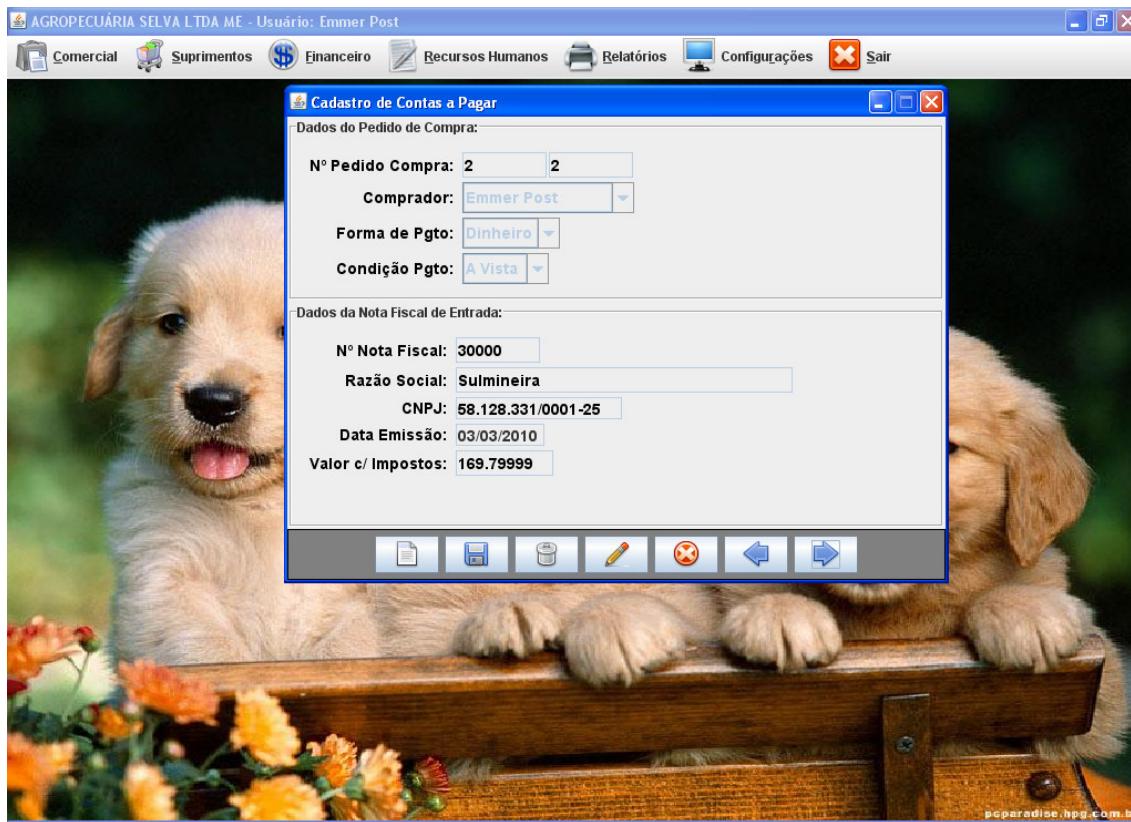


Figura 63: *Layout* cadastro de contas a pagar
Fonte: Guilherme Post

Layout de Lançamentos de vendas feitas com cartão.

Neste *layout* apresentado na figura 63 será possível após usuário após ter informado a condição de pagamento (cartão) e salvar a venda na tela de cadastro de vendas, lançar uma venda com cartão conforme a figura 78. Na parte inferior do *layout* possui os ícones de atalho para salvar cadastro, cancelar venda.

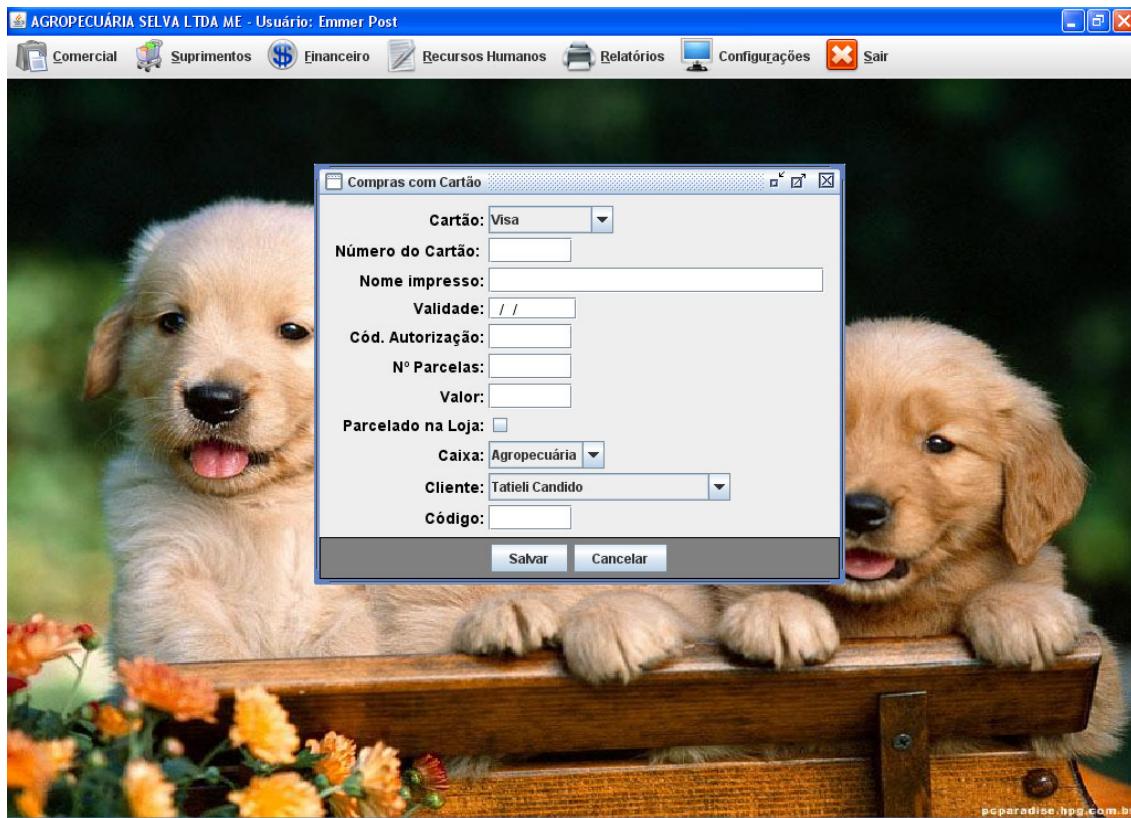


Figura 64: *Layout* de lançamentos de vendas feitas com cartão
Fonte: Guilherme Post

Layout de Forma de Pagamento e Condição de Pagamento.

Neste *layout* que é apresentado na figura 64 é possível o usuário realizar consultas, cadastro, alteração e exclusão de forma de pagamento assim como na tela condição de pagamento. Na parte inferior do *layout* possui os ícones de atalho. Acima dos ícones de atalho o usuário visualizar o código e descrição da forma de pagamento. Esta tela poderá ser solicitada pelo usuário do modulo do financeiro.

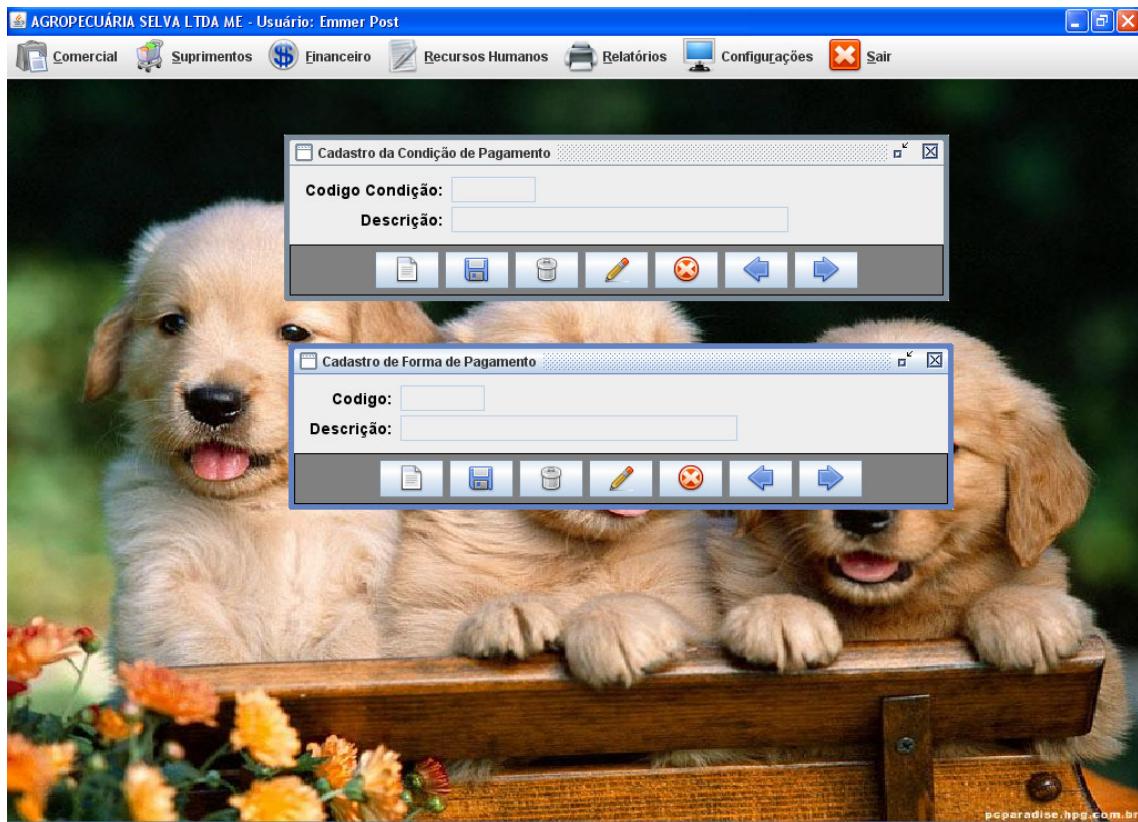


Figura 65: *Layout* cadastro de cadastro de forma de pagamento e condição de pagamento
Fonte: Guilherme Post

Layout de cadastro de Nota Fiscal de Entrada.

Neste *layout* apresentado na figura 65 é possível o usuário cadastrar a nota fiscal de entrada, realizar consultas, solicitar alteração do cadastro ou excluir uma nota fiscal. Este *layout* é o mesmo do cadastro de compras, para determinar que esta sendo cadastrada a NF, deverá ser informado o numero do pedido de compra que ira apresentar os dados da compra e em seguida deverá ser informada a data de entrega, se todos os itens conferidos com a NF e pedido de compra estiverem corretos deverá ser salvo o cadastro. Nesta tela é possível visualizar o nome do fornecedor e seus dados, o código e nome do comprador que emitiu a compra, os itens da compra, os valores totais e dados da compra.

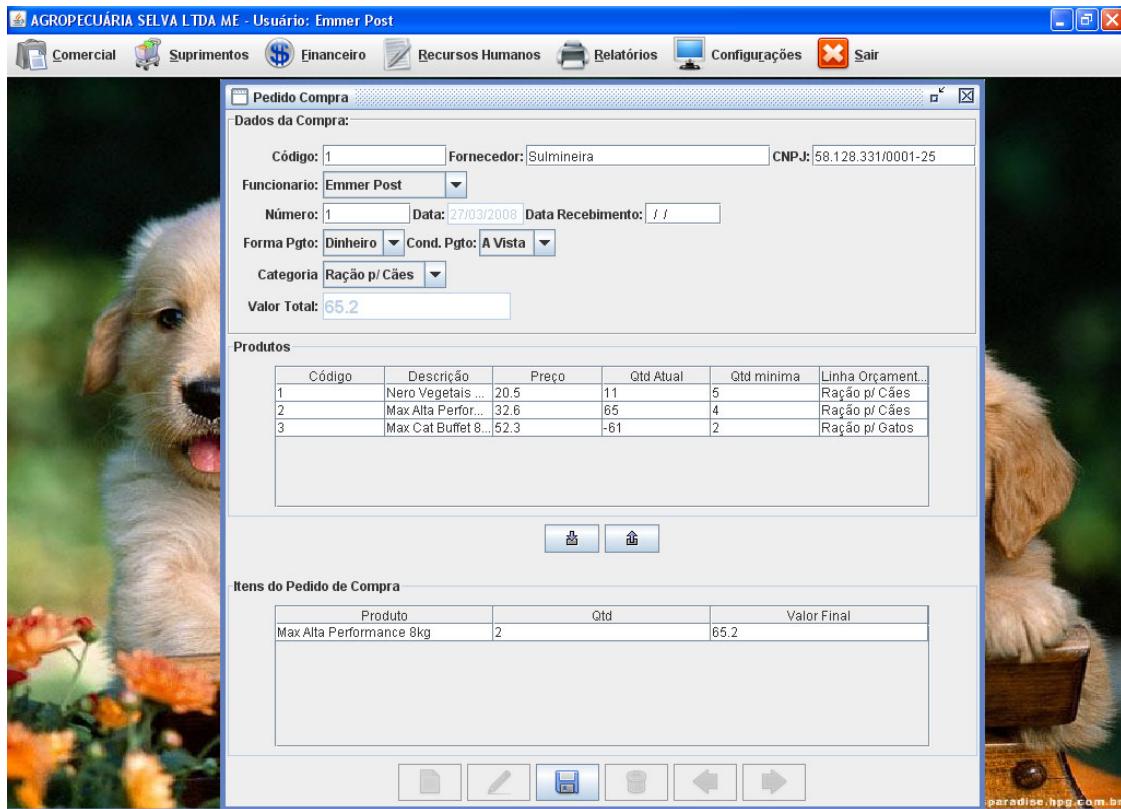


Figura 66: *Layout* cadastro de cadastro de nota fiscal de entrada
Fonte: Guilherme Post

Layout de cadastro de Produto.

Neste *layout* apresentado na figura 66 é possível o usuário realizar consultas, cadastro, alteração e exclusão de produtos. Na parte inferior do *layout* possui os ícones de atalho, para salvar, excluir, cadastrar ou alterar. Acima dos ícones de atalho o usuário visualizar o código, descrição do produto, quantidade atua, quantidade mínima deste produto, valor unitário, unidade de medida se é em pacote, avulso ou unitário e fornecedor que forneceu o produto.

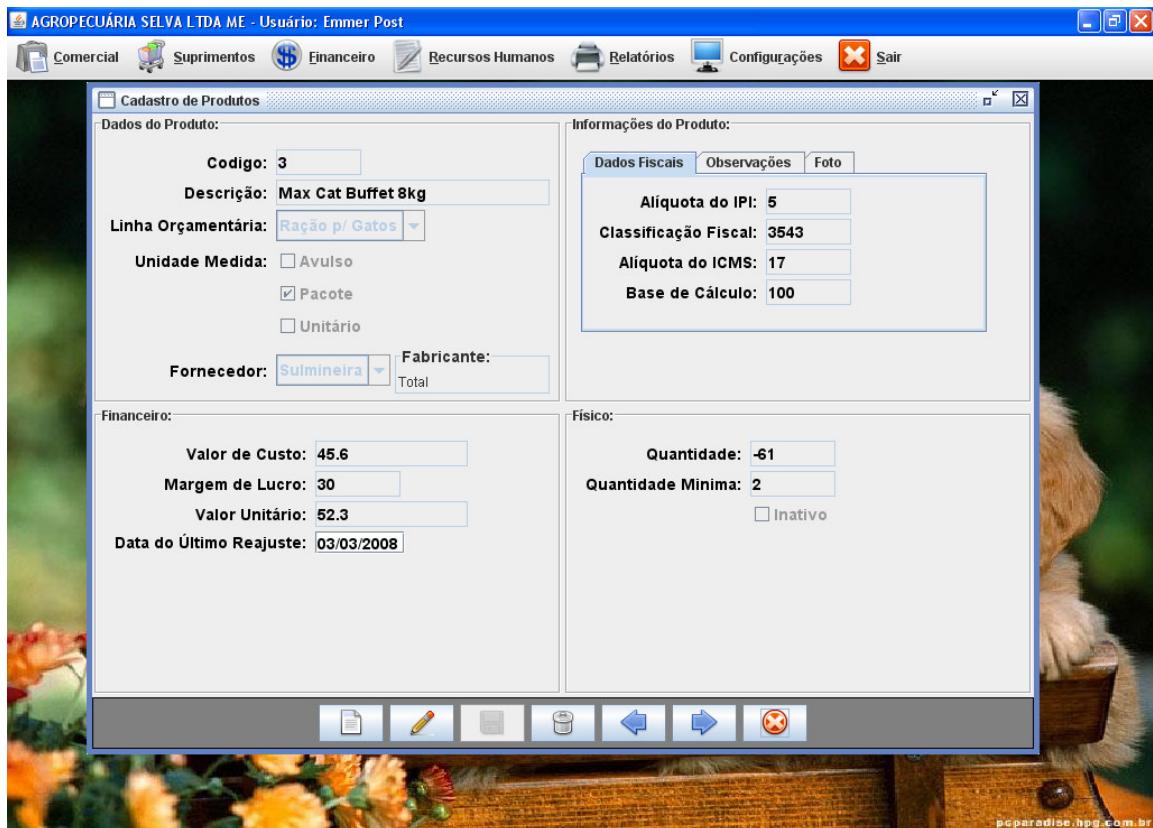


Figura 67: Layout cadastro de cadastro de produtos

Fonte: Guilherme Post

Layout de cadastro de Compra.

Neste *layout* apresentado na figura 67 é possível o usuário a cadastrar compra, realizar consultas, solicitar alteração do cadastro ou excluir uma compra. Na parte inferior do *layout* de compra possui os ícones de atalho para salvar cadastro, novo cadastro que é o cadastro de uma nova compra, excluir, consultar uma compra. Acima dos ícones de atalho o usuário poderá inserir ou ver os itens do pedido de compra. Nesta tela é possível visualizar a razão social do fornecedor e seus dados, o código e nome do comprador que emitiu a compra, os itens da compra, os valores totais e dados da compra.

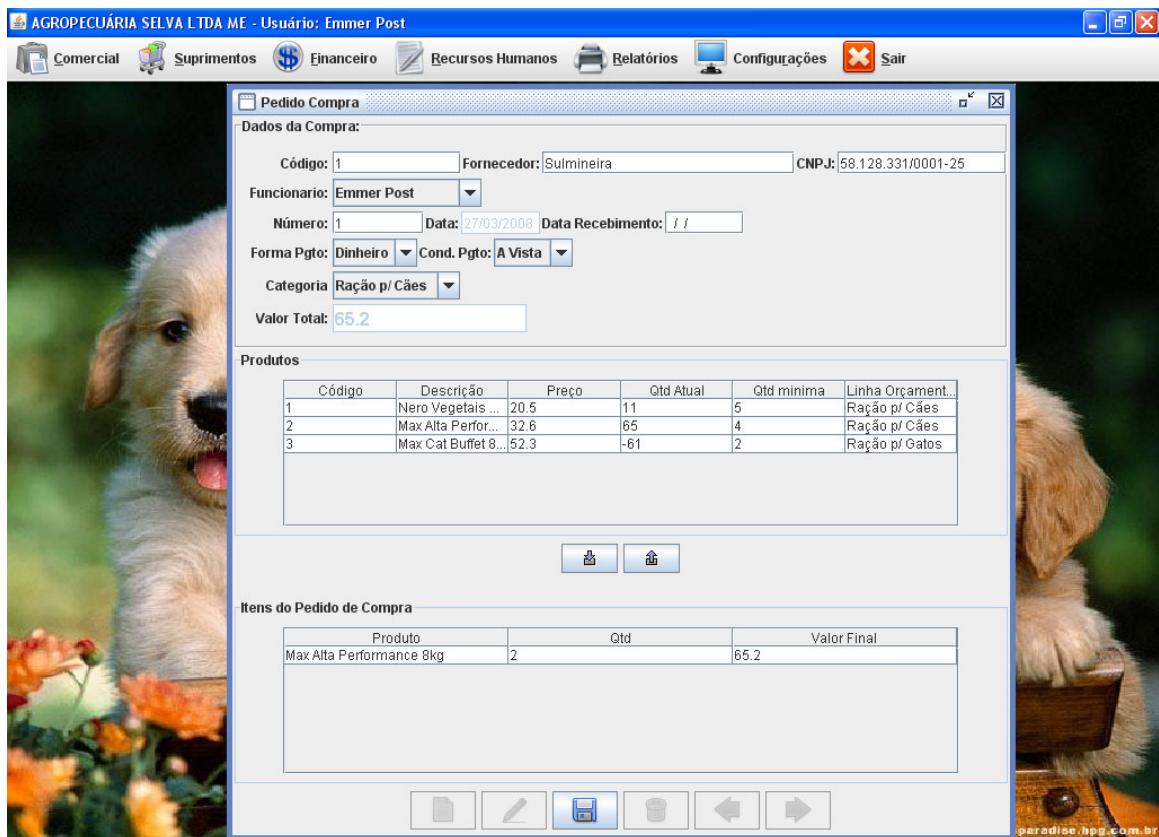


Figura 68: *Layout* cadastro de cadastro de compra

Fonte: Guilherme Post

Layout do cadastro de cartão.

Neste layout é possível cadastrar, consultar, excluir ou alterar os dados do cartão. É apresentado na figura 68.

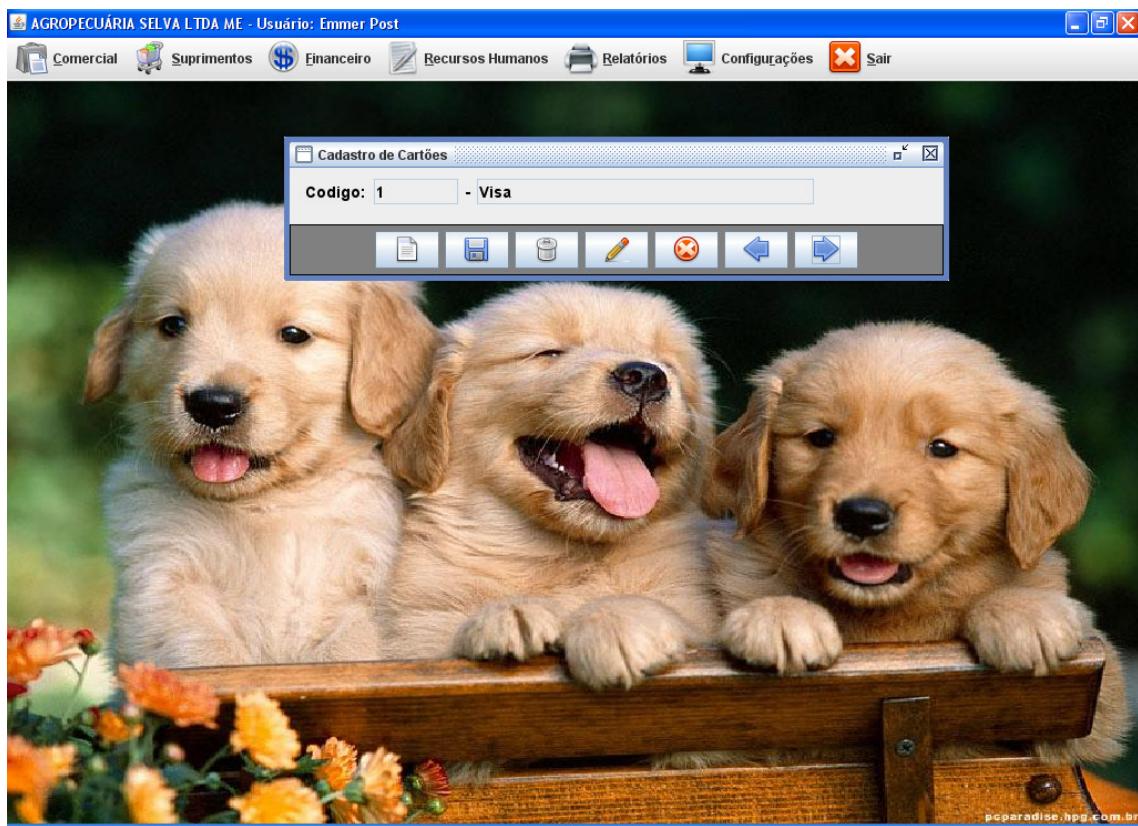


Figura 69: *Layout* cadastro de cartão
Fonte: Guilherme Post

Layout do cadastro de categorias.

Neste layout é possível cadastrar, consultar, excluir ou alterar os dados da categoria. É apresentado na figura 69.



Figura 70: *Layout* cadastro de categoria
Fonte: Guilherme Post

Layout do cadastro de fornecedores.

Neste layout é possível cadastrar, consultar, excluir ou alterar os dados dos fornecedores. É apresentado na figura 70.

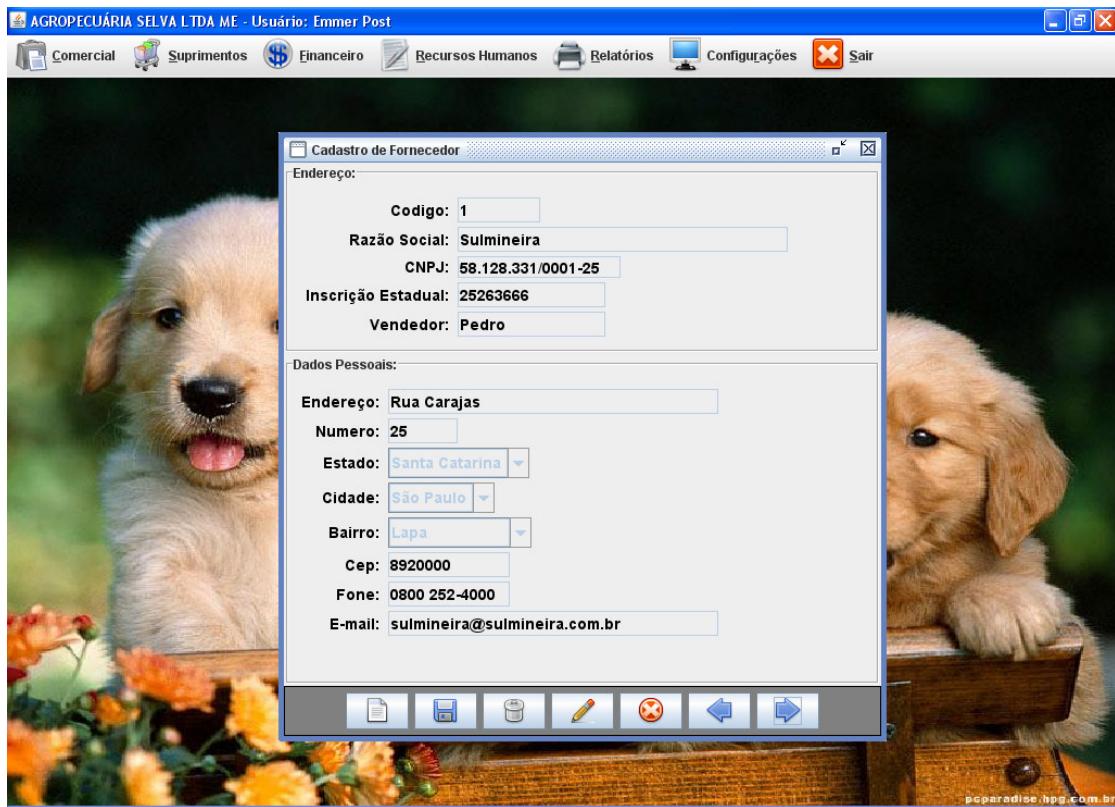


Figura 71: *Layout* cadastro de fornecedores
Fonte: Guilherme Post

Layout de consulta de clientes.

Neste layout é possível consultar, os dados dos clientes, na parte superior da tela são apresentados os filtros que poderão ser informados para realizar a consulta. É apresentado na figura 71.

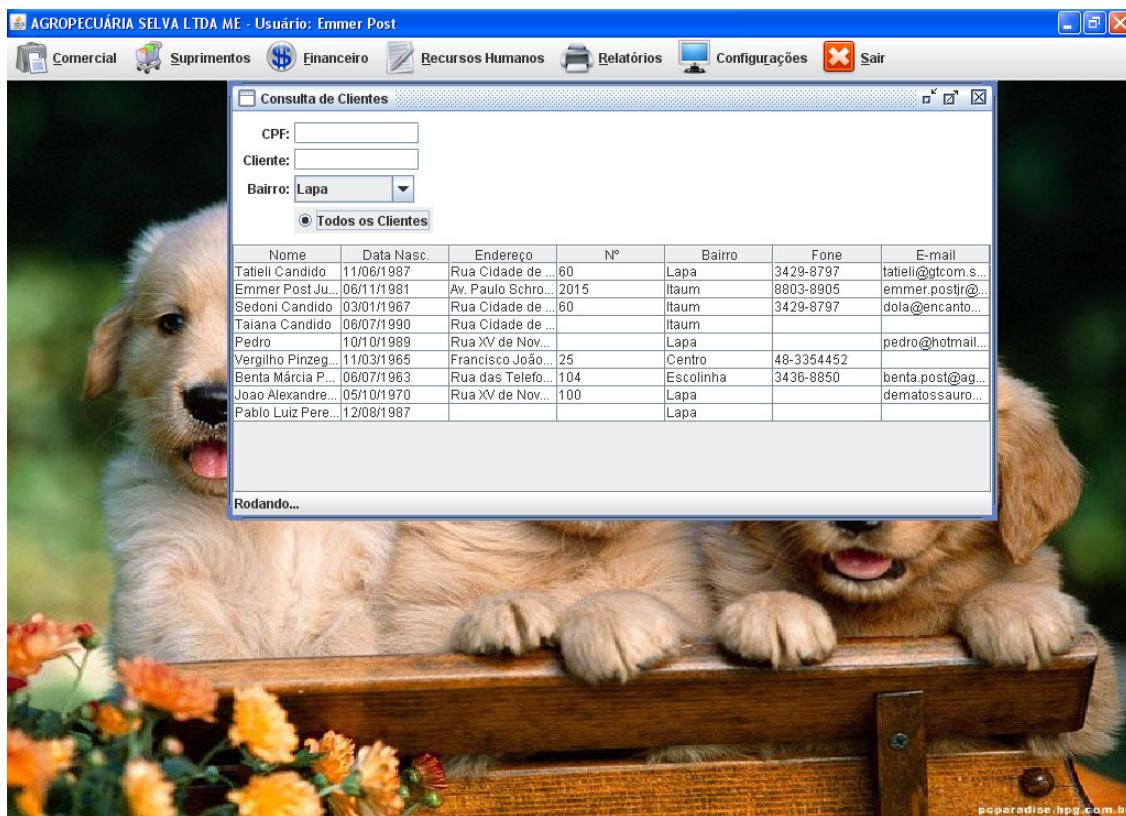


Figura 72: *Layout* consulta de clientes

Fonte: Guilherme Post

Layout de consulta de pedidos de compra.

Neste layout é possível consultar, os dados das compras realizadas aos fornecedores, na parte superior da tela são apresentados os filtros que poderão ser informados para realizar a consulta. É apresentado na figura 72.

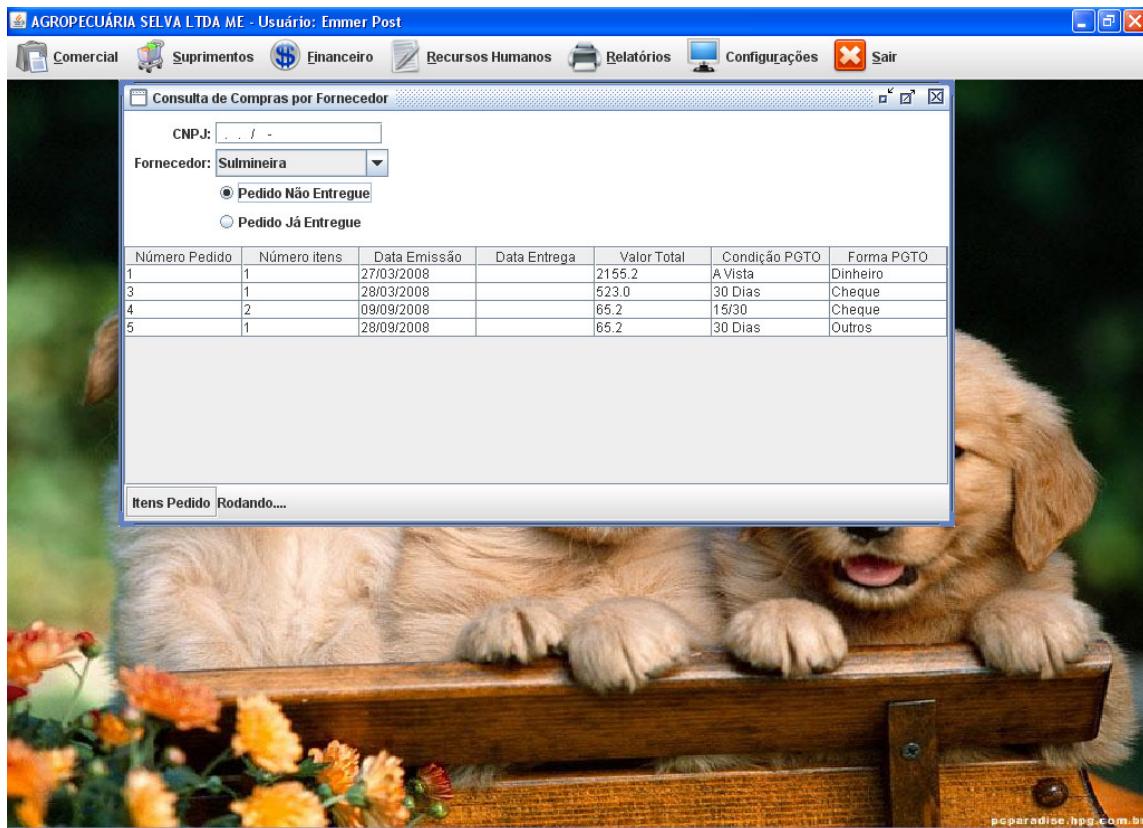


Figura 73: *Layout* consulta de pedidos de compra
Fonte: Guilherme Post

Layout de consulta de contas a pagar.

Neste layout é possível consultar, os dados das contas a pagar gerada através das compras realizadas aos fornecedores. É apresentado na figura 73.

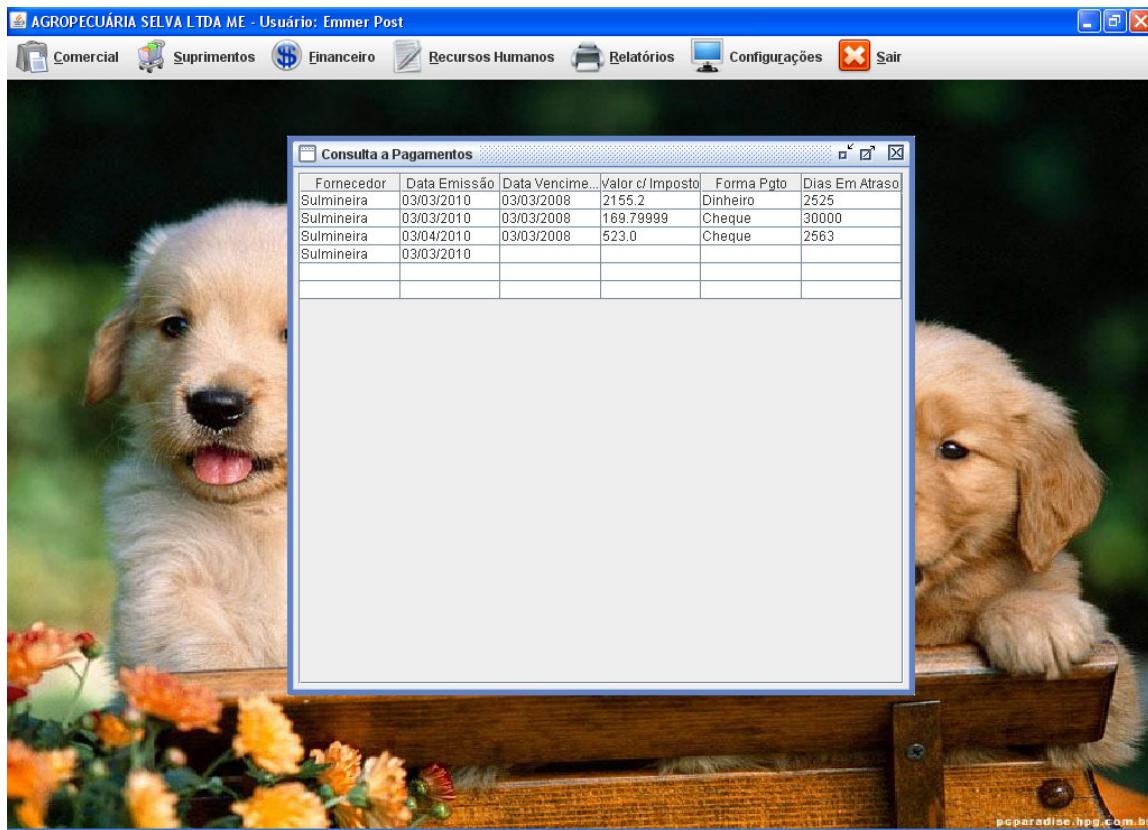


Figura 74: *Layout* consulta de contas a pagar
Fonte: Guilherme Post

Layout de consulta de contas a receber.

Neste layout é possível consultar, os dados das contas a receber gerada através das vendas realizadas pelos clientes. Nesta tela é possível visualizar os dados em tela ou solicitar o relatório das contas conforme parâmetros estipulados na tela. É apresentado na figura 74.

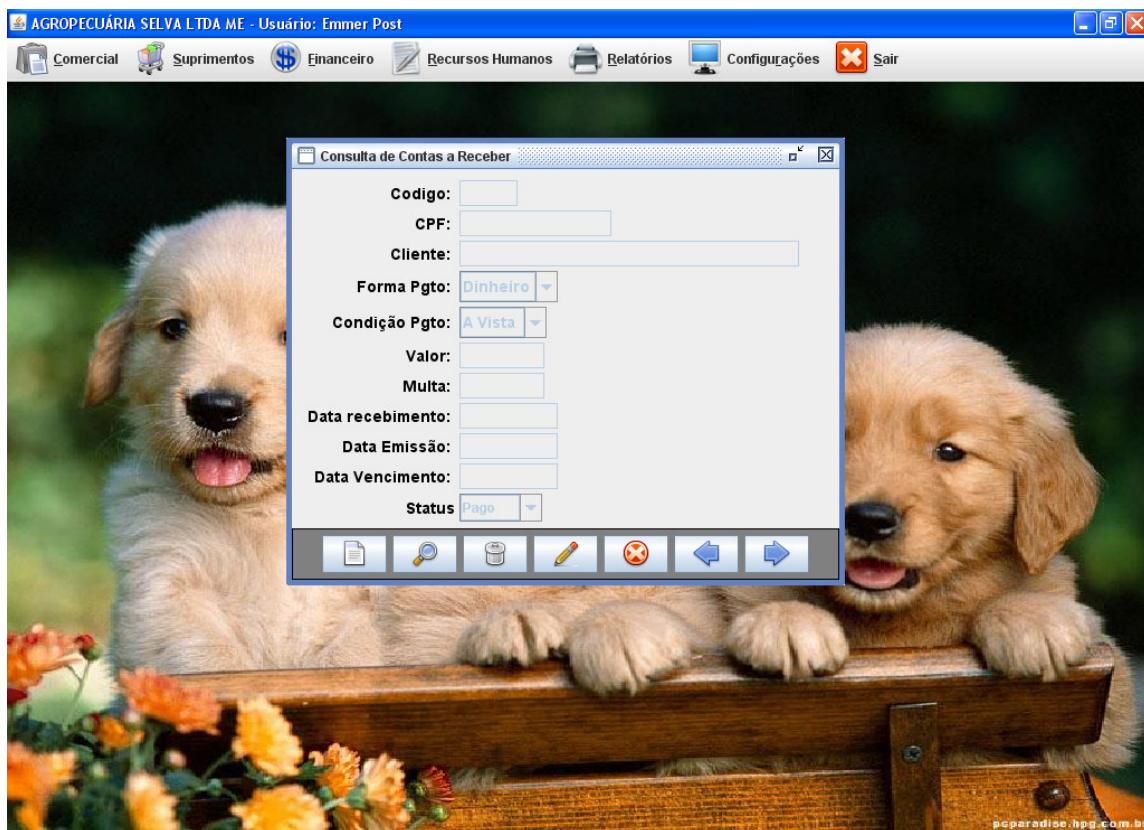


Figura 75: *Layout* consulta de contas a receber
Fonte: Guilherme Post

Layout de consulta de pedido de venda.

Neste layout é possível consultar, os dados das vendas realizadas pelos clientes. O usuário poderá informar os parâmetros da pesquisa em tela para filtrar a pesquisa. É apresentado na figura 75.

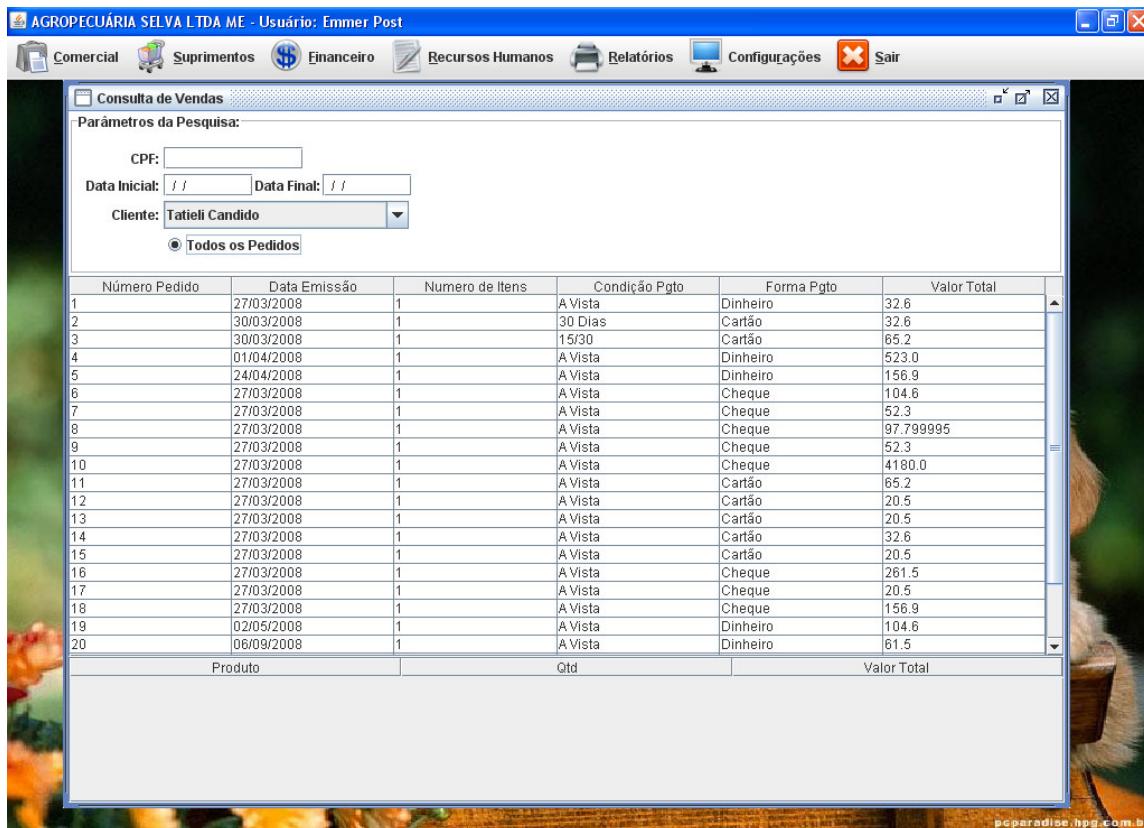


Figura 76: Layout consulta de vendas

Fonte: Guilherme Post

Layout de controle de vendas feitas com cartão.

Neste layout é possível consultar, os dados das vendas realizadas pelos clientes que solicitaram pagar com cartão. O usuário poderá informar os parâmetros da pesquisa em tela para filtrar a pesquisa. É apresentado na figura 76.

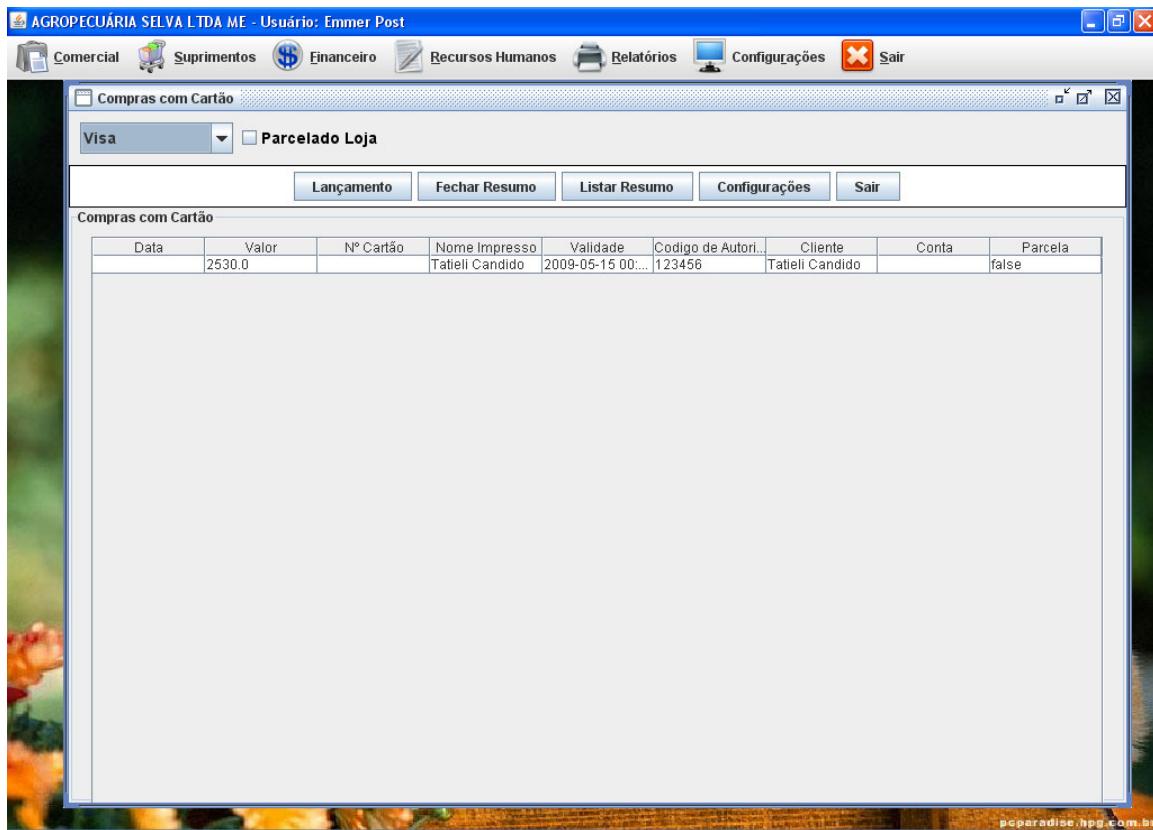


Figura 77: Layout controle de vendas feitas com cartão

Fonte: Guilherme Post

Layout de controle de vendas feitas com cheque.

Neste layout é possível consultar, os dados das vendas realizadas pelos clientes que solicitaram pagar com cheque. O usuário poderá informar os parâmetros da pesquisa em tela para filtrar a pesquisa. É apresentado na figura 77.

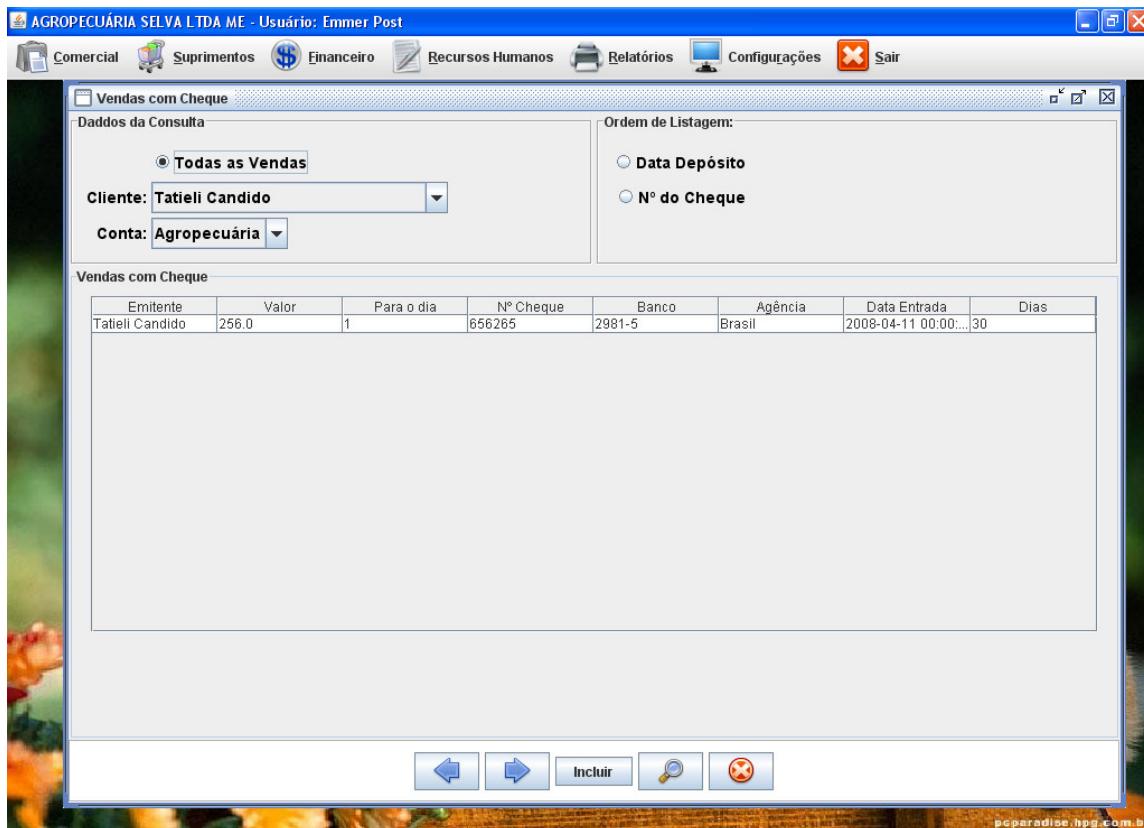


Figura 78: *Layout* controle de vendas feitas com cheque

Fonte: Guilherme Post

Layout de fluxo de caixa.

Neste layout é possível consultar, os dados das vendas e compras que geraram um contas a receber e a pagar. O usuário poderá informar os parâmetros da pesquisa em tela para filtrar a pesquisa. É apresentado na figura 78.

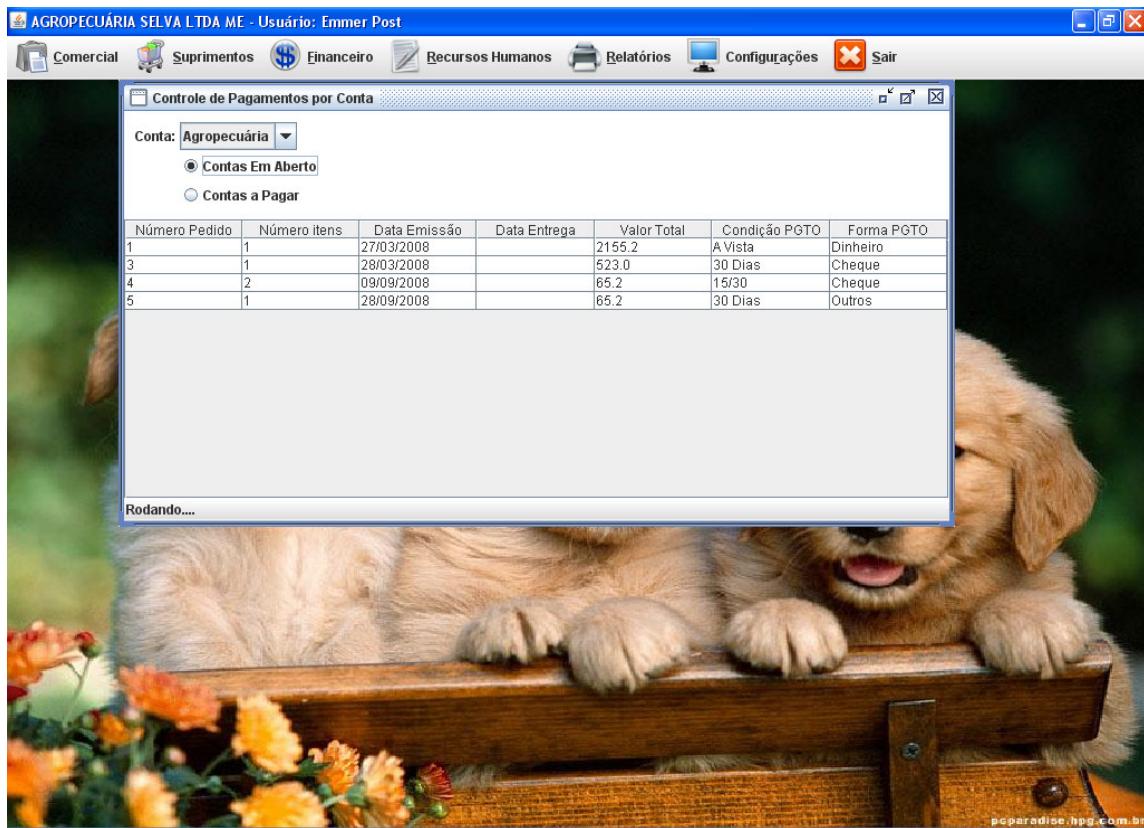


Figura 79: *Layout* de fluxo de caixa

Fonte: Guilherme Post

Layout que gera relatório de contas a pagar.

Neste layout é possível gerar um relatório de contas a pagar a partir dos parâmetros informados na tela que possibilitaram filtrar o relatório. O usuário poderá informar os parâmetros da pesquisa em tela para filtrar a pesquisa. É apresentado na figura 79.

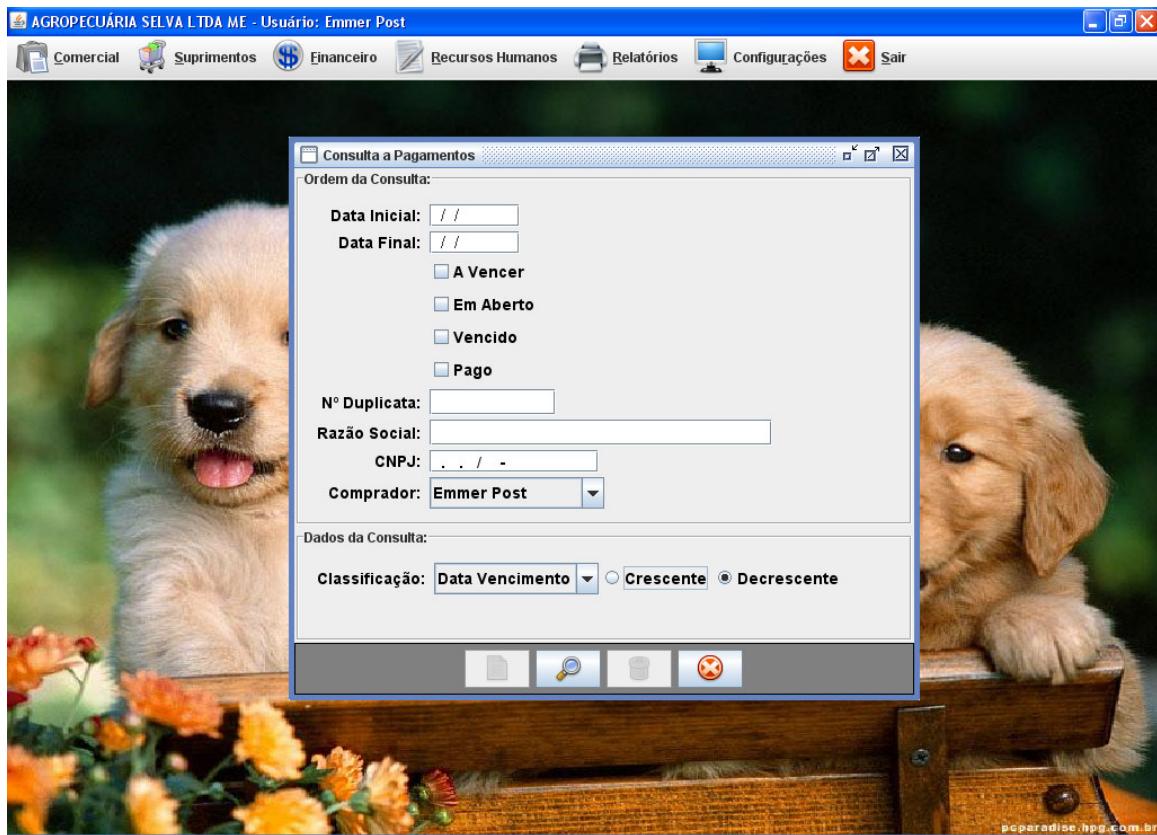


Figura 80: *Layout que gerar relatório de contas a pagar*

Fonte: Guilherme Post

Layout lançamento de vendas feitas com cheque.

Neste layout é possível lançar uma venda feita com cheque, possibilita digitar mais de uma folha de cheque para uma venda. O usuário poderá informar o numero de folhas de cheque e o valor total da venda e pressionar o botão salvar e informar os demais dados. É apresentado na figura 80.

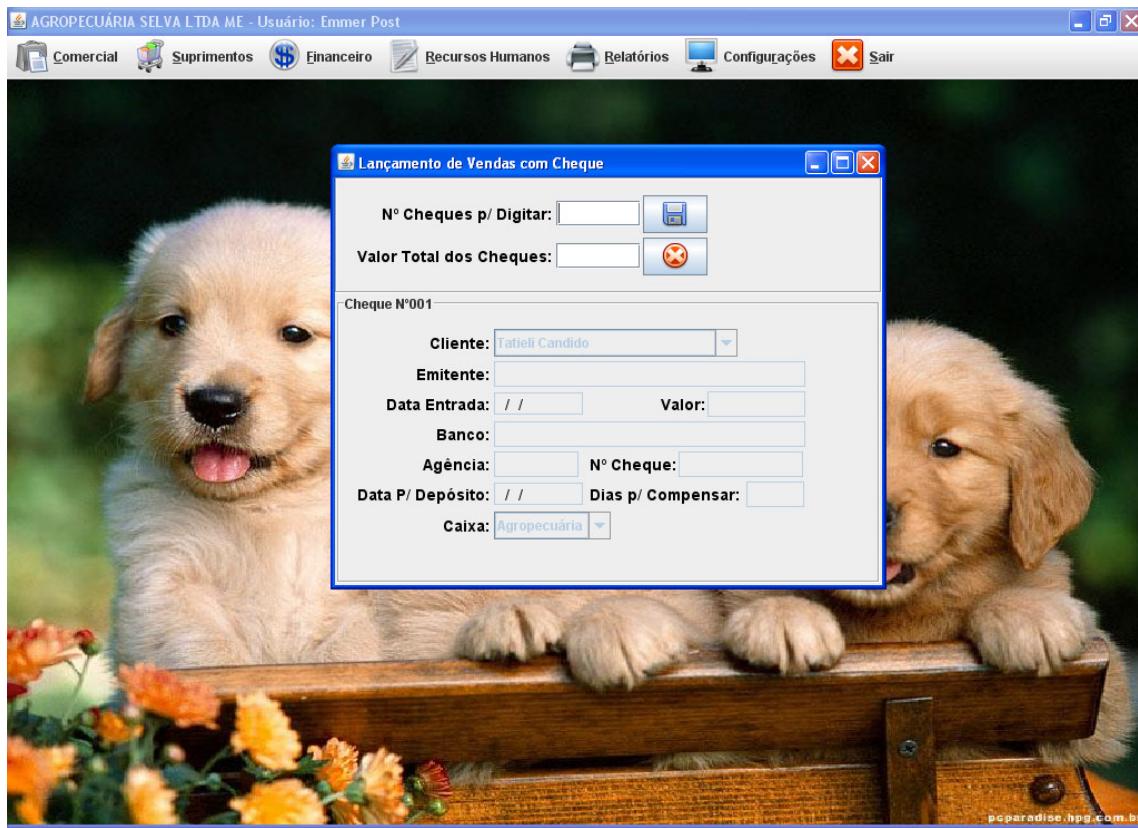


Figura 81: *Layout* lançamentos de vendas com cheque
Fonte: Guilherme Post

Layout reajuste de preço.

Neste layout é possível selecionar os itens em estoque e reajustar o preço conforme parâmetros informados na tela, após clicar no botão aplicar, será atualizado o preço em todas as vendas a prazo. É apresentado na figura 81.

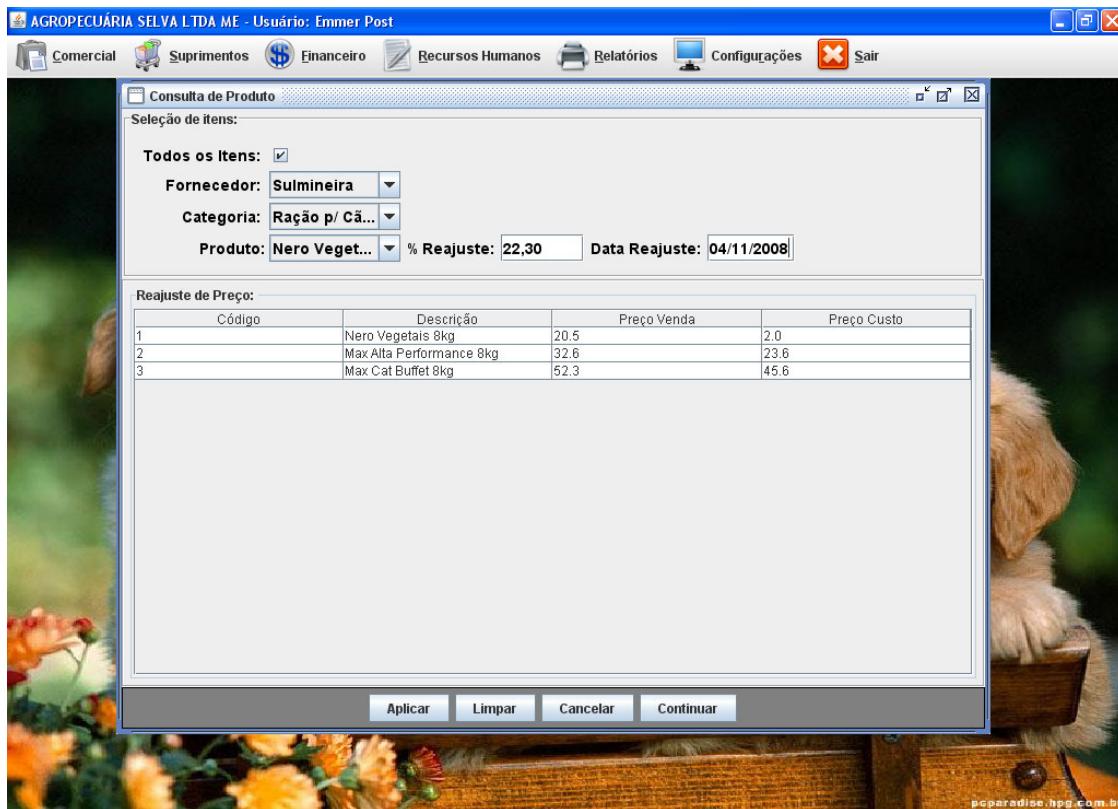


Figura 82: *Layout* reajuste de preço

Fonte: Guilherme Post

Layout venda resumida.

Neste layout é possível manter o cadastro de venda, utilizado para vendas com valores menores, onde a venda ocorre normalmente a vista, não havendo a necessidade de cadastrar o cliente e sim de controlar a saída de produtos e de entrada de caixa. É apresentado na figura 82.

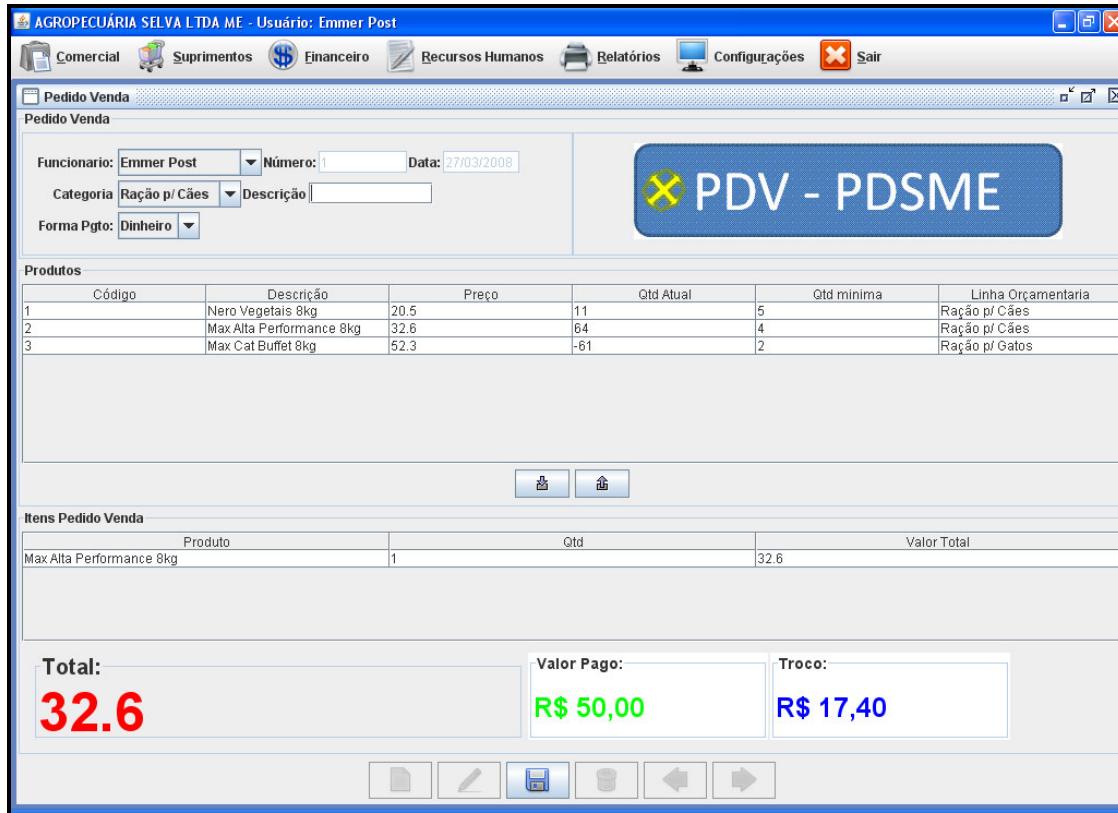


Figura 83: *Layout* venda resumida

Fonte: Guilherme Post

Layout do relatório de clientes.

Neste *layout* apresentado na figura 83 é possível o usuário analisar os dados dos clientes.

Relatório de Clientes					
Agropecuária Selva LTDA ME			Funcionário: Emmer Post		
Nome	Endereço	Nº	Bairro	Fone	E-mail
Benta Márcia Post	Rua das Telefonistas	104	Escolinha	3436-8850	benta.post@agroselva.com.br
Emmer Post Junior	Av. Paulo Schroeder	2015	Itaum	8803-8905	emmer.postjr@hotmail.com
Joao Alexandre Moreira de	Rua XV de Novembro	100	Lapa		dematossauro@ig.com.br
Pedro	Rua XV de Novembro		Lapa		pedro@hotmail.com
Sedoni Candido	Rua Cidade de Óros	60	Itaum	3429-8797	dola@encantoemagia.com.br
Taiana Candido	Rua Cidade de Orós		Itaum		
Tatieli Candido	Rua Cidade de Óros	60	Lapa	3429-8797	tatieli@gtcom.srv.br
Vergilho Pinzegher	Francisco João	25	Centro	48-3354452	

Figura 84: *Layout* do relatório de clientes,
Fonte: Guilherme Post

Layout do relatório de produtos em falta no estoque.

Neste *layout* apresentado na figura 84 é possível o usuário analisar os dados dos produtos, sendo possível identificar os produtos que deverão ser reposto no estoque. O usuário poderá através deste relatório montar seu pedido de compra, e solicitar aos fornecedores exatamente o que esta faltando, sem correr o risco de comprar produtos que já tenham um estoque considerável.

Relatório de Produtos				
Agropecuária Selva LTDA ME		Funcionário: Guilherme Post		
Data da Consulta: Dom, 5 Out 2008 22:27:46				
Código	Descrição	Qtd Atual	Qtd Mínima	Preço
1	Nero Vegetais 8kg	24	5	20,50
2	Max Alta Performance	111	4	32,60
3	Max Cat Buffet 8kg	-3	2	52,30

Figura 85: *Layout* do relatório de produtos

Fonte: Guilherme Post

Layout do relatório de contas a receber.

Neste *layout* apresentado na figura 85 é possível o usuário analisar os dados das contas a receber, sendo possível identificar os clientes que deverão quitar suas pendências até a data do vencimento. O usuário poderá através deste relatório identificar os clientes inadimplentes e realizar um controle sobre a decisão a ser tomada para evitar atrasos.

Relatório de Contas a Receber				
Agropecuária Selva LTDA ME 00.249.535/0001-42 Data da Consulta: Dom, 5 Out 2008 22:45:20				Funcionário: Emmer Post
Nome	Nº Venda	Forma Pgto	Data Vencimento	Valor Total
João Alexandre	00001	A Prazo	10/10/08	R\$ 32,00
	00009	A Prazo	10/10/08	R\$ 18,30
	00015	A Prazo	10/10/08	R\$ 10,60
Valor Total: R\$ 60,90				

Figura 86: *Layout* do relatório de contas a receber
 Fonte: Guilherme Post

Layout do relatório de contas a pagar.

Neste *layout* apresentado na figura 86 é possível o usuário analisar os dados das contas a pagar, sendo possível identificar as contas que deverão ser quitadas até a data do vencimento. O usuário poderá através deste relatório é possível identificar e controlar os pagamentos.

Relatório de Contas a Pagar				
Agropecuária Selva LTDA ME 00.249.535/0001-42			Funcionário: Emmer Post	
Data da Consulta: Dom, 5 Out 2008 23:02:21				
CNPJ	Fornecedor	Forma Pgto	Data Vencimento	Valor Total
58.128.331/0001-26	Sulmineira S/A	Duplicata	30/10/08	R\$ 3.256,36
36.105.333/0001-96	Casa do Adubo	Cheque	26/10/08	R\$ 8.363,00

Figura 87: Layout do relatório de contas a pagar

Fonte: Guilherme Post

3.8. Implementação

Esta etapa é responsável pela codificação dos dados coletados após o levantamento e análise dos requisitos. A programação do sistema ocorre a partir deste ponto, pois neste momento está estabelecido onde queremos chegar, o objetivo já está definida no projeto, documentado e acordado com os usuários do que é preciso que o sistema faça.

Com o intuito de programar visando sempre seguir a metodologia para futuras manutenções obterem um maior sucesso, o sistema foi desenvolvido através das técnicas de *Design Patterns* (Padrão de Projeto). Conforme SAMPAIO (2007) o padrão de projeto representam uma técnica excelente para reduzir os custos de manutenção de aplicativos, através do privilégio da Composição sobre a Herança.

Seguindo este padrão de projeto as manutenções futuras deste sistema, ficará menos complexas, devido redução do acoplamento entre componentes, mesmo quando o sistema é

robusto consegue se tornar de fácil manutenção, deixando os desenvolvedores se concentrar nas regras de negócio e não mais na lógica intermediária afirma SAMPAIO (2007).

O padrão de projeto a ser seguido para o desenvolvimento desse sistema é apresentado da seguinte forma, baseado na literatura de SAMPAIO (2007):

- Apresentação: padrões relacionados à interface com o usuário;
- Negócios: padrões que executam ou distribuem a execução de regras de negócio;
- Integração: Padrões que integram componentes diversos, remotos ou não.

No entanto foi escolhido os padrões EJB (*Enterprise JavaBeans*) para desenvolver o sistema. Conforme SAMPAIO (2007) *Enterprise JavaBeans* é uma tecnologia embutida na plataforma Java 2 *Enterprise Edition* para desenvolver componentes distribuídos.

Os tipos de EJB's segundo SAMPAIO (2007) são:

Session Beans: representam as regras de negócio ou transações que o sistema deve realizar.

Entity Beans: representam as entidades do banco de dados instanciadas como objetos, são as entidades na qual os *Session Beans* agem.

Message-driven Beans: são os componentes mais utilizados para processar mensagens assíncronas. Onde é enviada uma mensagem ao *Session bean* (uma regra de negócio, exemplo consultar cliente), que irá acessar um *Entity Bean* (uma tabela do banco de dados, exemplo Cliente).

O projeto foi dividido em quatro pacotes (*Control, Forms, Model, DAO, DAOFactory*).

DAO significa *Data Access Object*, é um padrão de projeto utilizado para criar uma camada de persistência responsável por realizar a integração entre a aplicação e o banco de dados.

Forms é a interface que o usuário interage, são as telas do sistema, responsável por criar uma máscara entre usuário e banco de dados, que facilita o entendimento do que se deve fazer na hora de cadastrar uma venda, cadastrar um cliente, SAMPAIO (2007).

Model é um modelo, criado para ligar a interface e a DAO, esse modelo serve para apresentar os valores atuais as Interfaces, SAMPAIO (2007).

DAOFactory é o gerenciador das entidades, conhecido como fábrica de objetos conforme SAMPAIO(2007).

Controls recebe a solicitação da *forms*, invoca as regras de negócios necessárias, comanda a alteração do estado do Model e a atualização das informações exibidas na *forms*, SAMPAIO (2007).

Para conexão com o banco de dados o provedor JPA (Java Persistence API) utilizado é o *Hibernate Entity Manager*. O Hibernate é uma ferramenta de mapeamento objeto/relacional para Java. Ela transforma os dados tabulares de um banco de dados em um grafo de objetos definido pelo desenvolvedor. Usando o Hibernate, o desenvolvedor deixa de escrever muito do código de acesso a banco de dados e de SQL que ele escreveria não usando a ferramenta, acelerando a velocidade do seu desenvolvimento de uma forma considerável.

Os *layouts* de relatórios foram desenvolvidos utilizando a ferramenta iReport na versão 2.0. O iReport é uma ferramenta que visa facilitar a construção de relatórios utilizando a biblioteca JasperReports (<http://jasperreports.sourceforge.net>) através de uma interface gráfica desenvolvida em Swing. Ele dispõe de importantes ferramentas para desenvolver relatórios complexos e demorados.

A implementação iniciou-se pelos casos de uso priorizados na tabela 34. A figura 87 apresenta o código fonte de controle da tela de emissão de compras quando pressionado botão salvar.

```

if(comp.getName().equals("btnSalvar")){
    Compra pedAtual = modelo.getMeuPedido();
    pedAtual.setcomp_oid(frame.getcomp_oid ());
    pedAtual.setcomp_dt_emissao(frame.getcomp_dt_emissao());
    pedAtual.setcomp_dt_receb(frame.getcomp_dt_receb ());
    pedAtual.setcomp_cnpj(frame.getcomp_cnpj());
    pedAtual.setcomp_razao(frame.getcomp_razao());
    pedAtual.setform_oid(frame.getform_oid());
    pedAtual.setcond_oid(frame.getcond_oid());
    pedAtual.setcat_oid(frame.getcat_oid());
    pedAtual.setfunc_oid(frame.getfunc_oid());
    pedAtual.setcomp_vlr_tot(frame.getcomp_vlr_tot ());
    List<Itemc> colItens = frame.getModelTabItens().getColItens();
    for(Itemc it:colItens){
        Produto pro = it.getProduto();
        pro.setQtd(pro.getQtd() + it.getQuantidade());
    }
    pedAtual.setColItens(colItens);
    DAOFactory.getInstance().getPedCDAO().salvarPedido(pedAtual);
    frame.habObjetos();
}

```

Figura 88: Código fonte da classe de controle de compras
Fonte: Guilherme Post

Na figura 89 é apresentado o diagrama de interface, mostrando os pacotes *Action*, *Forms* e *Model*, para facilitar o entendimento deste diagrama foi exposto apenas a o processo manter cadastro de cliente.

No pacote *Action*, possui CadClienteAction, nesta classe é armazenado os métodos (ou regras de negócios) que são acionadas através da FrmCadCliente que esta dentro do pacote *Forms*.

O pacote *Forms* possui a interface FrmCadCliente que serve para interagir com o usuário. Na seqüência da figura é apresentado logo abaixo da *Forms* o pacote *Models*, nele é encontrado todos os atributos relacionados na frmCadCliente, que servira de conexão para o pacote Dao apresentado na figura 90. As classes do banco são apresentadas dentro do pacote DAO, contem o clienteDAO que deve realizar a integração entre a aplicação e o banco de dado, DAOFactory é o gerenciador das entidades. Na figura 88 é apresentada a tabela do banco chamada de entity.



Figura 89: Diagrama de Persistência, Cliente Entity.
Fonte: Guilherme Post

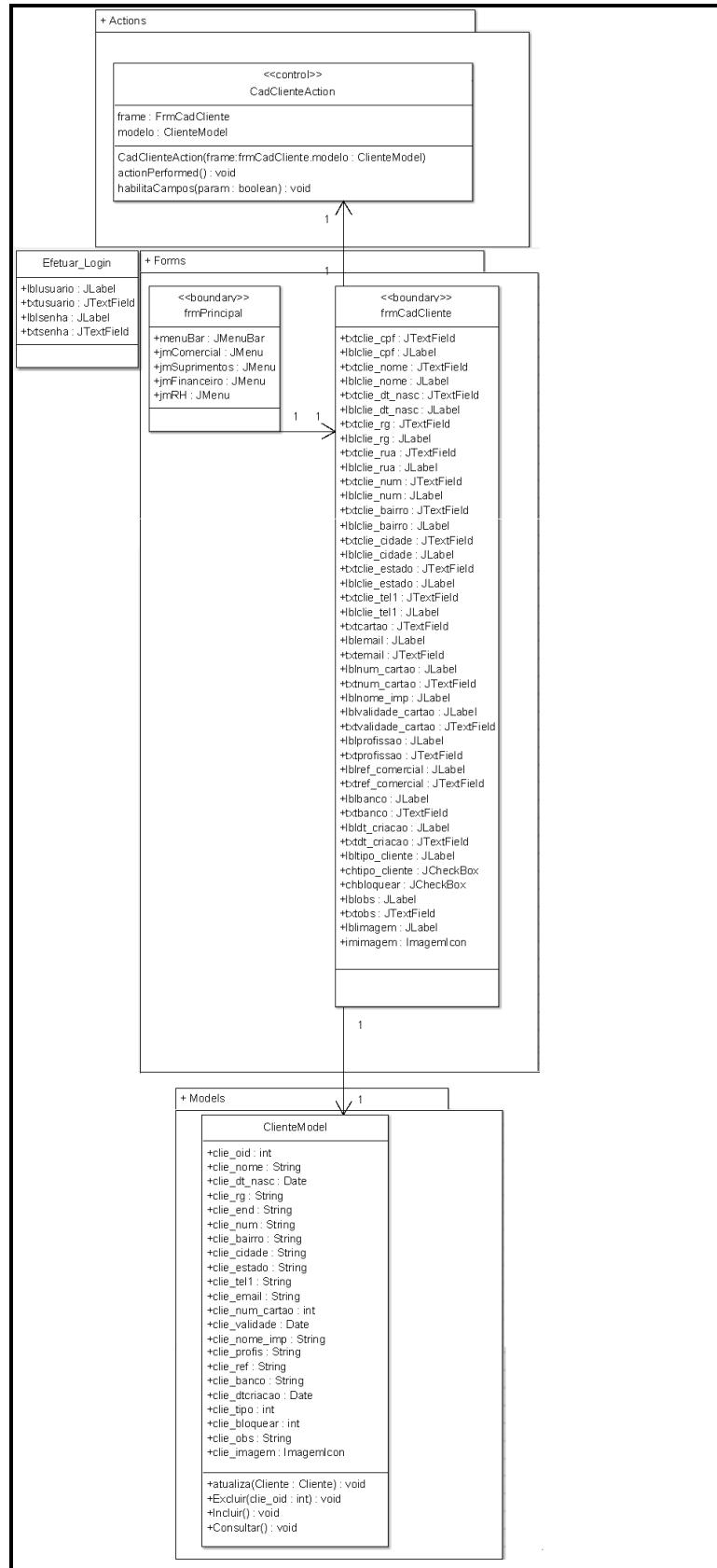


Figura 90: Diagrama de Interface.

Fonte: Guilherme Post

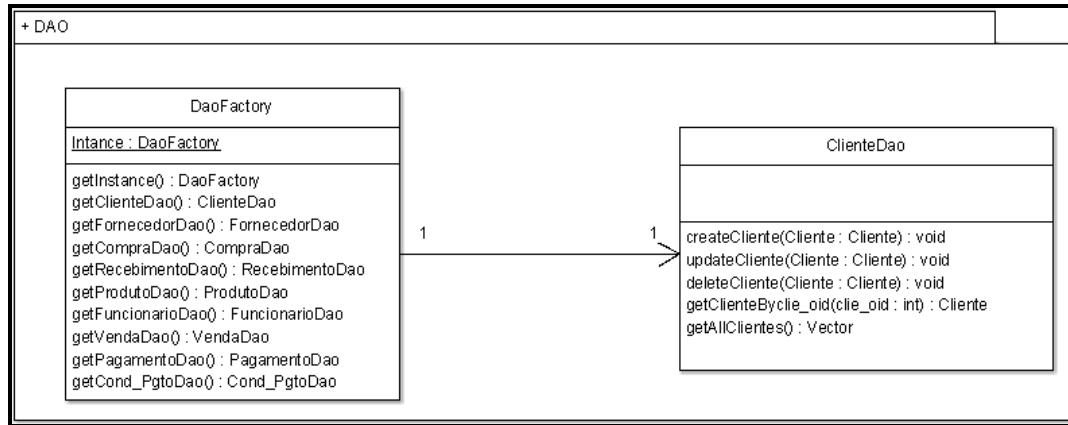


Figura 91: Diagrama de Persistência, DAOFactory e ClienteDAO.

Fonte: Guilherme Post

3.9. Testes

Os testes realizados ocorrem durante a finalização do desenvolvimento de cada programa, ao finalizar um programa (Exemplo cadastro de clientes) são feitos os testes funcionais, de usabilidade, de lógica, carga, recuperação, também é analisada a aparência do sistema no intuito de identificar se os campos estão nos tamanhos e posições que facilitem o entendimento do que se deve fazer nesta tela.

Os testes são realizados de maneira a forçar o erro, são informados valores para identificar se o sistema está calculando certo, é feito o cadastro da venda ou de compra, para identificar a baixa ou entrada de produtos no estoque. Esta etapa é fundamental para a aceitação do sistema na empresa, minimizando ou eliminando o risco do sistema chegar ao cliente com problemas.

Os testes unitários ocorreram de forma a verificar:

- Lógica interna dos programas, analisando as estruturas dos códigos;
- Troca correta de informação entre as interfaces;
- Dados armazenados temporariamente e sua integridade durante a execução;
- Testes de condição/limites;
- Erros e validações;
- Padrões e normas.

Esses tipos de testes auxiliam e muito o desenvolvedor a possibilitando criar ambiente de testes, ainda tornando o desenvolvimento do sistema confiável, com qualidade e diminuindo ou eliminando as chances de implantar um sistema com defeitos.

Os testes de integração funcionalidades foram realizados durante todo o desenvolvimento. Exemplo foram os testes feitos nas telas que consistem em identificar se os campos possuem identificação, se a escrita esta correta, se as lógicas dos botões funcionam, assim como as funcionalidades de cada um, verificando se ocorre a integração entre os módulos e se de fato esta gravando as informações corretas nas tabelas do banco de dados.

Os botões possuem ícones padronizados, para facilitar o entendimento da sua funcionalidade, cada campo em tela possui um *ToolTipText* que apresenta uma mensagem que informa o que deve ser informado naquele campo no momento em que o usuário passa com o *mouse* por cima do campo, e cada campo possui um *label* que identifica o nome do campo.

Durante o processo na qual esta se esta cadastrando um novo cliente, os botões (editar, novo, excluir, consultar o próximo e o anterior) devem estar desabilitados. Na figura 92 é apresentada uma tela com as possíveis verificações que são testadas.

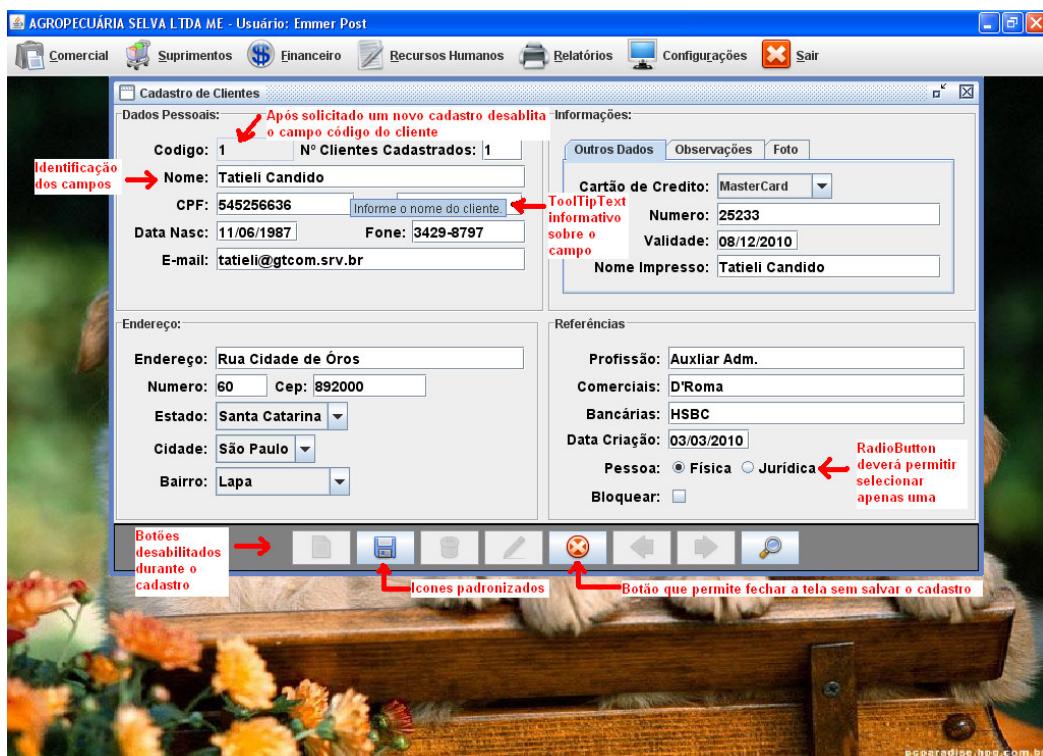


Figura 92: Classe de testes TesteConsulta.

Fonte: Guilherme Post

3.10. Implantação

A implantação ocorrerá após a implementação do sistema, nesta etapa é destinada uma pessoa, normalmente o consultor para realizar a implantação do sistema na empresa. Deverá ser instalado sistema nas máquinas dos usuários, logo após a instalação do *software* deverá ser instalado o banco de dados no servidor e fazer a conexão do sistema com o banco de dados.

O sistema será implantado em apenas uma máquina disponibilizada pela empresa, neste computador possui as seguintes especificações:

Hardware:

- Processador - DualCore Intel Pentium D 805, 2666 MHz (20 x 133);
- Placa Mãe – Chipset VIA P4M800 PRO;
- Memória do sistema – 1024mb;
- HD – 120 gb;
- Adaptador gráfico – NVIDIA Geaforce4 MX 4000 (64mb);
- Adaptador de som - *Realtek ALC655 @ VIA AC'97 Enhanced Audio Controller*

Software:

- Antivírus – Nod32 versão 2.51.20;
- Sistema Operacional: Microsoft Windows XP Profissional;

Neste computador serão instalados:

- Banco de dados - SQL Server 2005;
- J2SE Development Kit (JDK) 1.6;
- iReport;

Durante todo o processo de implantação haverá dúvidas dos usuários com relação ao uso do sistema, para sanar as dúvidas o consultor disponibilizará um cronograma de treinamento para capacitar os usuários, apresentando os módulos, as funções, e a forma de utilização. Com este treinamento os usuários poderão ter pleno conhecimento do sistema, suas funções e utilidades, visando à aceitação do sistema pelos os usuários.

Neste terceiro e último capítulo foi possível apresentar como foi desenvolvido o sistema, foram apresentados através de uma forma metodologia através de cada etapa

(Levantamento de Requisitos, Análise, Projeto, Implementação, Testes e Implantação) o uso de representações gráficas as funcionalidades do sistema, possibilitando atingir os objetivos propostos no Pré Projeto.

CONSIDERAÇÕES FINAIS

O trabalho de conclusão de estágio foi desenvolvido na empresa Agropecuária Selva LTDA, teve como tema o desenvolvimento de um sistema para gerenciar os processos realizados pela empresa. Foram realizadas entrevistas com os usuários para verificar e analisar os problemas enfrentados atualmente, tendo como objetivo desenvolver um sistema para facilitar e resolver alguns problemas enfrentados.

Com o sistema desenvolvido neste projeto para gerenciar os processos, possibilitará o cliente a ter um controle gerencial do que a empresa está vendendo, comprando, pagando e recebendo. Sendo assim, o cliente poderá tomar decisões administrativas e logísticas, com como e onde deverá investir, otimizando custos e estoque, ou até auxiliar nas vendas à clientes inadimplentes.

Para desenvolver este sistema foi seguido cronograma apresentado no projeto e as etapas foram concluídas dentro dos prazos estabelecidos. Utilizou-se da metodologia UML, que apresenta de uma forma mais detalhada através de seus diagramas, modelos o que realmente é preciso desenvolver para atender as necessidades do cliente.

Durante o projeto percebeu-se que para desenvolver um sistema de qualidade, seguro, eficiente e que atenda os requisitos do cliente é preciso ter entendimento pleno da funcionalidade dos processos realizados pela empresa, bem como, conhecer área de negócios o que é fundamental para se desenvolver um sistema.

O projeto foi implantado através de um sistema piloto, onde será disponibilizado o sistema em um computador para os usuários se familiarizarem com o sistema. Durante este período, serão realizados treinamentos dos usuários para sanar as dúvidas e orientá-los sobre os procedimentos a serem efetuados para o uso do sistema.

Com este trabalho, foi possível enriquecer o conhecimento e a prática do uso da metodologia do Processo Unificado com o emprego da linguagem de modelagem de dados a UML. Que possibilitou ter uma visão panorâmica do sistema, apresentando através das representações gráficas e tabelas, uma forma mais clara e objetiva das informações adquiridas e objetivos propostos durante o levantamento de requisitos, análise, projeto, implementação, testes e implantação, assim como também conhecer a área de negócios a qual a empresa atua.

REFERÊNCIAS

ALBERTÃO, Sebastião Edmar. **ERP sistemas de gestão empresarial: metodologia para avaliação, seleção e implantação.** São Paulo: IGLU, 2001.

BEZERRA, Eduardo. **Princípios de análise e projeto de sistemas com UML.** Rio de Janeiro: Campus, 2002.

BOGGS, Wendy; BOGGS, Michael. **Mastering: a bíblia UML com rational rose 2002.** Rio de Janeiro: Alta Books, 2002.

BOOCH, G; JACOBSON, I.; RUMBAUGH, J. **UML: guia do usuário.** Rio de Janeiro: Campus. 2000.

CONSO, Pierre; POULAIN, Pierre. **Informática na Administração.** Rio de Janeiro: Ao Livro técnico, 1972.

DEITEL, H. M; DEITEL, P. J. **Java, como programar.** 6. ed. São Paulo: Prentice Hall, 2005.

DATE, C. J. **Introdução a sistemas de Banco de Dados.** 7ed. Rio de Janeiro: Campus, 2000.

DAVIS, Willian. **Análise e projeto de sistemas: uma abordagem estruturada,** 1986.

FURGERI, Sergio. **Java 2: ensino didático: desenvolvendo e implementando aplicações.** São Paulo: Érica, 2002.

GUEDES, Gilleanes T.A. **UML: uma abordagem prática.** São Paulo: Novatec, 2004.

KORTH, Henry F; SILBERSCHATZ, Abraham. **Sistema de bancos de dados.** São Paulo: McGraw-Hill, 1989.

LAUDON, Kenneth C.; Laudon, Jane Price. **Gerenciamento de sistemas de informação.** Rio de Janeiro: LTC, 1999.

LEÃO, Renata de Oliveira; SILVA, João Carlos da. **SQL server 2000: estrutura e implementação de sistemas de banco de dados.** São Paulo: Érica, 2002.

LEE, Richard C.; Tepfenhart, Willian M. **Guia Prático de Desenvolvimento Orientado a Objeto.** São Paulo: Makron, 2005.

LIA. **Introdução a Modelo Relacional.** Disponível em:
<<http://www.lia.ufc.br/~eti2005/menu/modulos/BDR/BDR-ModeloRelacional.pdf>>. Acesso em 31 jul. 2008, 00h31min.

MATOS, Alexandre Veloso de. **UML: prático e descomplicado.** São Paulo: Érica, 2002.

MELO, Ana Cristi. **Desenvolvendo Aplicações com UML,** Rio de Janeiro: Brasport, 2005.

MULLER, Robert. **Projeto de Banco de Dados. Usando UML para modelagem de dados.** São Paulo: Berkely, 2002.

OLIVEIRA, Jayr Figueiredo. **Metodologia para desenvolvimento de projeto de sistemas.** 3^a ed. São Paulo: Érica, 1999.

PRESSMAN, Roger S. **Engenharia de software.** São Paulo: Makron Books, 1995.

PILONE, Dan; PITMAN, Neil. **UML 2: rápido e prático .** Rio de Janeiro: Alta Books, 2006.

REZENDE, Denis A. **Engenharia de Software e Sistemas de Informação.** Rio de Janeiro: Brasport, 1999.

_____. **Engenharia de software e sistemas de informação.** Rio de Janeiro: Brasport, 2005.

_____. **Sistemas de informações organizacionais.** Rio de Janeiro: Brasport, 2005.

REZENDE, Denis Alcides; ABREU, Aline França de. **Tecnologia da informação aplicada a sistemas de informação empresariais: o papel estratégico da informação e dos sistemas de informação nas empresas.** São Paulo: Atlas, 2000.

REZENDE, Denis Alcides. **Planejamento de sistemas de informação e informática: guia prático para planejar a tecnologia da informação integrada ao planejamento estratégico das organizações.** São Paulo: Atlas, 2003.

SAMPAIO, Cleuton de Melo Jr. **Guia do Java: Enterprise Edition 5: desenvolvendo aplicações corporativas.** Rio de Janeiro, Brasport, 2007.

SCOTT, Kendall. **Processo Unificado Explicado.** Porto Alegre: Bookman, 2003.

SILVA, José Pereira. **Analise financeira das empresas.** 2^a ed. São Paulo: Atlas, 1990.

SILBERSCHATZ, Abraham. **Sistemas de banco de dados.** São Paulo, Makron Books, 1999.

SLACK, Nigel et al. **Administração da produção: edição compacta.** São Paulo: Atlas, 2006.

SOMMERVILLE, Ian. **Engenharia de Software.** São Paulo: Addison Wesley, 2003.

STAIR, Ralph M. **Princípios de sistemas de informação: uma abordagem gerencial.** 2 ed. Rio de Janeiro: JC, 1998

TURBAN, Efraim; RAINER JR., Rex Kelly; POTTER, Richard E. **Administração da tecnologia da informação : teoria e prática.** 2.ed. Rio de Janeiro: Campus, 2003.

YOURDON, Edward. **Análise Estruturada Moderna.** Rio de Janeiro: Campus, 1992.