UNIVERSIDADE DO PLANALTO CATARINENSE
DEPARTAMENTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE INFORMÁTICA
(BACHARELADO)

PROTÓTIPO DE UM SISTEMA PARA GERÊNCIA DE PIZZARIAS

Relatório do Trabalho de Conclusão de Curso submetido à Universidade do Planalto Catarinense para obtenção dos créditos de disciplina com nome equivalente no curso de Informática - Bacharelado.

ANGELO CORBELLINI MALTY DE ANDRADE

LAGES, DEZEMBRO DE 2002

UNIVERSIDADE DO PLANALTO CATARINENSE
DEPARTAMENTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE INFORMÁTICA
(BACHARELADO)

PROTÓTIPO DE UM SISTEMA PARA GERÊNCIA DE PIZZARIAS

Relatório do Trabalho de Conclusão de Curso submetido à Universidade do Planalto Catarinense para obtenção dos créditos de disciplina com nome equivalente no curso de Informática - Bacharelado.

ANGELO CORBELLINI MALTY DE ANDRADE

Orientador(a) : Wilson Castello Branco Neto, M Eng.

LAGES, DEZEMBRO DE 2002

PROTÍTIPO DE UM SISTEMA PRA GERÊNCIAS DE PIZZARIAS

ANGELO CORBELLINI MALTY DE ANDRADE

ESTE RELATÓRIO, DO TRABALHO DE CONCLUSÃO DE CURSO, FOI JULGADO ADEQUADO PARA OBTENÇÃO DOS CRÉDITOS DA DISCIPLINA DE TRABALHO DE CONCLUSÃO DE CURSO DO VIII SEMESTRE, OBRIGATÓRIA PARA OBTENÇÃO DO TÍTULO DE:

BACHAREL EM INFORMÁTICA

Prof. Wilson Castello Branco Neto,
M Eng.
Orientador

BANCA EXAMINADORA:

Prof. Alexandre Perin deSouza.
UNIPLAC

Prof. João Oliveira Neto.
UNIPLAC

Prof. Angelo Augusto Frozza, Esp.
Supervisor de TCC

Prof. Alexandre Perin de Souza
Coordenador de Curso

Lages, 13 de Dezembro de 2002 (26/11/2002)

SUMÁRIO

LISTA DE ILUSTRAÇÕES ..... **VI**
RESUMO ..... **VII**
ABSTRACT ..... **VIII**
1. INTRODUÇÃO **1**
1.1. APRESENTAÇÃO **4**
1.2. DESCRIÇÃO DO PROBLEMA **5**
1.3. JUSTIFICATIVA **5**
1.4. OBJETIVOS **6**
1.4.1. Objetivo Geral 6
1.4.2. Objetivos Específicos 6
1.5. METODOLOGIA **6**
2. O PROCESSO DE DESENVOLVIMENTO DE SISTEMAS **8**
2.1. INTRODUÇÃO **8**
2.2. ANÁLISE E ESPECIFICAÇÃO DE REQUISITOS **8**
2.2.1. Dificuldades no Processo de Extração de Requisitos 9
2.2.2. Técnicas para Extração de Requisitos 10
FIGURA 2.1 - PROTOTIPAÇÃO. **14**
2.3. PROJETO (UML) **15**
2.3.1. A Linguagem UML 15
2.3.2. Diagramas da UML 17
2.3.3. Diagrama de Robustez 18
2.4. IMPLEMENTAÇÃO E TESTES **19**
3. PROJETO DE BANCO DE DADOS **20**
3.1. INTRODUÇÃO **20**
3.2. VANTAGENS DE UM SISTEMA GERENCIADOR DE BANCO DE DADOS **20**
3.3. ABSTRAÇÃO DOS DADOS **22**
3.4. PESSOAS ENVOLVIDAS EM UM BANCO DE DADOS **22**

3.4.1. O Administrador 22
3.4.2. Usuários Finais 23
3.5. DIAGRAMAS ENTIDADE-RELACIONAMENTO 24
3.5.1. Entidades e Atributos 24
3.5.2. Relacionamento 25
3.5.3. Especialização e Generalização 26
3.5.4. Herança de Atributos 27
3.6. CONSIDERAÇÕES FINAIS 28
4. MODELAGEM DO SISTEMA 29
4.1. INTRODUÇÃO 29
4.2. DOMÍNIO DO PROBLEMA 29
4.3. ESPECIFICAÇÃO DE REQUISITOS E ANÁLISE DE ROBUSTEZ 30
4.3.1. Diagramas de Use-Case e Robustez 30
4.3.2. Especificação do Sistema 30
4.3.3. Descrição de Cenários 31
4.4. DIAGRAMA DE SEQÜÊNCIA 37
5. APRESENTAÇÃO DO SISTEMA 44
6. CONCLUSÃO 51
7. REFERÊNCIAS 53

LISTA DE ILUSTRAÇÕES

FIGURA 2.1. Prototipação..... 11
FIGURA 3.1. Os três níveis de abstração de dados 20
FIGURA 3.2. Conjunto de entidade cliente e seus atributos 22
FIGURA 3.3. Relacionamento muitos para muitos 23
FIGURA 3.4. Generalização e Especialização 24
FIGURA 4.1. Diagrama de Use-Case 27
FIGURA 4.1. Diagrama de Use-Case 27
FIGURA 4.2. Diagrama e Robustez – cadastro de produtos 29
FIGURA 4.3. Diagrama e Robustez – verifica relatório 30
FIGURA 4.4. Diagrama e Robustez – cadastro de ingredientes 31
FIGURA 4.5. Diagrama e Robustez – cadastro de funcionários 32
FIGURA 4.6. Diagrama e Robustez – controle de pedidos 34
FIGURA 4.7. Diagrama e Seqüência – cadastro de produtos 35
FIGURA 4.8. Diagrama e Seqüência – verifica relatório 36
FIGURA 4.9. Diagrama e Seqüência – cadastro de ingredientes 37
FIGURA 4.10. Diagrama e Seqüência – cadastro de funcionários 38
FIGURA 4.11. Diagrama e Seqüência – controle de pedidos 39
FIGURA 4.12. Diagrama Entidade / Relacionamento 40
FIGURA 5.1. Tela Principal do Sistema 42
FIGURA 5.2. Tela para Cadastro de Pizzas 42
FIGURA 5.3. Tela para Efetuar Pedidos 43
FIGURA 5.4. Tela para Consultar Pedidos 44
FIGURA 5.5. Tela para Consultar Produtos 45
FIGURA 5.6. Tela para Relatório de Bebidas 45
FIGURA 5.7. Tela para Gerar Conta 46
FIGURA 5.8. Tela Pizzas mais Vendidas 47
FIGURA 5.9. Tela de Ajuda 47

RESUMO

O presente trabalho apresenta um sistema que visa agilizar o processo de gerência de uma pizzaria. O objetivo é cadastrar os produtos, ingredientes, pratos e efetuar os pedidos e gerar as contas referentes a cada mesa. Além disto o sistema provê módulos para consultas diversas e geração de relatórios, para uma melhor visualização das informações. As telas do sistema são de fácil operação visando uma melhor interação com o usuário.

Para o desenvolvimento do mesmo serão seguidas algumas etapas, visando diminuir a quantidade de erros e facilitar a manutenção futura do sistema.

A definição destas etapas é um assunto que tem sido longamente tratado por diversos autores, entretanto nota-se uma grande diversidade de opiniões. De maneira geral são basicamente três as etapas mais importantes: análise e especificação de requisitos; projeto; implementação e testes. A primeira etapa foi executada através de entrevistas com o cliente e análise do sistema já existente, sendo os requisitos levantados especificados através de diagramas de use-case ou casos de uso.

Para o projeto do sistema foram utilizados os diagramas de seqüência e robustez, e para o projeto do Banco de Dados foi criado um diagrama de Entidade / Relacionamento. A implementação foi realizada utilizando um ambiente de programação Borland Delphi e o gerenciador de Banco de Dados Interbase.

Palavras-chave: Análise e especificação de requisitos; Projeto; Implementação; Banco de Dados; Sistema para Gerência de Pizzarias.

ABSTRACT

The present work presents a sistem that intend to make faster the process of the management of a pizza house. The objective is to register in cadastre the products, ingredients, dishes and effectivate order and generate the bills of each table. Besides that, provides research sistem for several consults and report generator to a better view of informations. The sistem screens are easy to opperate, aiming a better interaction with the user.

To the development of that will be followed some stages, looking for mistakes deduction and to make easier the future maintenance of the system.

The stages definition is a subject that has wridely treated by severals authors, however it's noticed a great diversity of opinions, but basicly there are three important stages: analysis and requeriment especification; project; implement and tests. The first stage was executed by interviewing the client and analysis of the already exsisting system, and the raised requirement especified by dagrams of use-case.

To the project of the system was used the system diagrams and stunght, and for the project of Database was criated a Entity / Relationship diagram. The project was placed use in a Borland Delphi program enviroment and the database Interbase manager.

Palavras-chave: Analysis and requeriment especification; Project; Implement; Database; Pizzas house management system.

1. INTRODUÇÃO

1.1. Apresentação

Um sistema de informação, nos dias de hoje, é um fator fundamental em uma organização de grande e pequeno porte. Estas empresas há alguns anos atrás, passaram por uma fase difícil em relação à informática, pois os altos custos dos equipamentos tornavam esta idéia completamente impraticável para as mesmas. Ainda hoje os sistemas de informação são caros, principalmente para empresas de pequeno porte. (SILVA, 1994)

Com o passar do tempo, esta dificuldade foi mudando, os avanços tecnológicos, a chegada dos microcomputadores tornaram então, esta idéia mais atraente para as empresas. (SILVA, 1994)

No Brasil, a abertura de mercado foi um fator importante para o desenvolvimento das empresas. Além disso, os meios de comunicação mais eficazes permitiram que estas trocassem informações entre si, e também estivessem sempre atualizadas. Novas tecnologias estão, cada vez mais, emergindo no mercado, tornando a informática um grande diferencial para as empresas em um mercado tão competitivo como o atual. Um sistema automatizado, agiliza os processos e também permite um melhor controle sobre os negócios.

Uma pizzaria, apesar de ser um pequeno negócio, pode utilizar sem dúvida um sistema computacional para agilizar tarefas e melhorar suas atividades.

Antes que um sistema seja implementado, é necessária a modelagem do mesmo. A linguagem UML é importante, porque descreve o sistema sob diferentes pontos de vista, mostrando desde a forma de interação entre o usuário e o sistema, através de diagramas Use-Case, até a seqüência dos eventos ocorridos, bem como os

objetos envolvidos nas diferentes transações, entre outras características. Estes diagramas auxiliam no momento da implementação deixando a estrutura do sistema mais clara para o desenvolvedor.

Uma modelagem bem elaborada, prevendo as situações que venham a acontecer no futuro, deixa o sistema menos sujeito a erros, facilitando o restante do processo de desenvolvimento.

1.2. Descrição do Problema

Neste trabalho, será enfocado o desenvolvimento de um sistema capaz de gerenciar as informações de uma pizzaria, visando torná-la mais eficiente e competitiva. Como parte importante para a resolução deste problema, serão estudados conceitos relativos a Análise e Projeto de Sistemas e projeto de Banco de Dados, para que o sistema resultante seja confiável e atenda as necessidades do cliente.

1.3. Justificativa

Por menor que seja um empreendimento, uma gerência eficiente pode ser o diferencial entre ele e a concorrência. Devido a isto, as empresas estão cada vez mais buscando ferramentas computacionais que as auxiliem na execução de tarefas, agilizando seus processos e tornando-as mais eficientes.

O sistema resultante deste trabalho será utilizado por uma pizzaria, que atualmente tem problemas devido à falta de um sistema de gerência eficiente, entre eles: a demora para fechar os pedidos (causando insatisfação nos clientes), a dificuldade no controle de estoque, que leva a perda de produtos (validade vencida) ou a falta de produtos, por não se ter uma idéia do que é gasto.

Atualmente a empresa em questão possui um sistema com uma interface de difícil manuseio, entre outros problemas.

1.4. Objetivos

1.4.1. Objetivo Geral

Este trabalho tem como objetivo informatizar o controle de uma pizzaria. Será considerado desde a movimentação do estoque de produtos até os pedidos dos clientes. Além disso, será desenvolvido um módulo para a geração de relatórios para que se possa analisar o andamento do negócio, de diferentes pontos de vista.

1.4.2. Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Estudar os conceitos relacionados a Desenvolvimento de Sistemas e Banco de Dados.
- Identificar as atividades que são executadas na empresa em questão.
- Analisar o sistema utilizado atualmente na empresa.
- Modelar o sistema de acordo com os requisitos levantados acima.
- Implementação e testes de um sistema que atenda as necessidades previamente identificadas.

1.5. Metodologia

Para que os objetivos sejam alcançados, este trabalho será desenvolvido de acordo com um embasamento teórico em projeto de Banco de Dados, Análise, Projeto, Implementação de Sistemas e UML (linguagem de modelagem unificada).

Paralelo ao estudo destes temas, será feito também o levantamento das necessidades da empresa através de entrevistas e observações “in loco” para o levantamento de requisitos.

Com base nos conceitos previamente estudados e requisitos levantados, será feita a modelagem da estrutura de Banco de Dados e do funcionamento do sistema. A

implementação do sistema será feita em Delphi junto com o Banco de Dados interbase, objetivando um controle geral de pedidos até o fechamento dos mesmos.

Os testes serão realizados durante a implementação simulando as situações que possam ocorrer em suas operações, para que o sistema fique menos propenso a erros na hora de sua finalização.

2. O PROCESSO DE DESENVOLVIMENTO DE SISTEMAS

2.1. Introdução

No meio em que se vive hoje, é muito usada a palavra sistema. É comum ouvir falar em sistema (respiratório, nervoso, sistema de folha de pagamento, sistema bancário entre outros), por isso é difícil conceituar com precisão o termo.

"Visto de uma forma genérica, podemos definir sistema como um conjunto de partes que interagem, visando um objetivo específico".(SILVA, 1994, p.05)

A análise de requisitos é a primeira etapa do desenvolvimento de um sistema. Está fase é importante, pois é nela que são levantadas as informações sobre o problema do cliente e a solução que ele espera, falhas nesta etapa podem levar a problemas na eficiência do sistema.

2.2. Análise e Especificação de Requisitos

A análise de requisitos é extremamente importante para o processo de desenvolvimento de sistemas, porque ela é a base para o sistema como um todo. Se a mesma estiver incorreta, o sistema será ineficiente, levando à insatisfação do usuário e causando aborrecimentos à empresa e ao desenvolvedor.

A pessoa responsável pela análise de requisitos é o Analista de Sistemas. Ele se preocupa em obter e organizar da melhor maneira possível as informações do cliente.

Do mesmo modo que estas informações chegam, o aprimoramento das mesmas torna-se importante para o funcionamento do sistema. Existem etapas que o analista deve seguir na análise de requisitos, são elas:

1) *Reconhecimento do Problema.*
2) *Avaliação e Síntese.*
3) *Modelagem.*
4) *Especificação.*
5) *Revisão para Testes.*

No reconhecimento do problema o analista se depara com o sistema a ser desenvolvido. Na avaliação e síntese, as análises são vistas junto com o cliente, e devido a isto perguntas são feitas como "*quais as funções o sistema terá para a execução"* e *"quais interfaces serão definidas"*.

Na modelagem do sistema é colocado em prática todas as avaliações e sínteses usando ferramentas *case*, e após esta etapa a especificação é realizada de acordo com as funções de desempenho do sistema.

Com base nisto a revisão para testes é feita junto ao cliente e ao desenvolvedor.

2.2.1. Dificuldades no Processo de Extração de Requisitos

O processo de extração de requisitos pode tornar-se difícil, tanto por problemas associados ao desenvolvedor quanto por problemas com o cliente. Os principais problemas que ocorrem na extração de requisitos são:

1) Falta de conhecimento do cliente com relação às suas necessidades. Isto se deve ao fato de que os usuários possuem uma vaga noção do que verdadeiramente precisam e do que o produto pode lhes oferecer, durante o levantamento dos requisitos.

2) Falta de conhecimento do desenvolvedor sobre o domínio do problema, levando-o a tomar decisões erradas e/ou precipitadas.

3) O desenvolvedor toma todas as decisões sobre o sistema sem aceitar idéias opostas às dele. Desta forma acaba ficando um clima insatisfatório pela falta de interação com os outros desenvolvedores.

4) Comunicação inadequada entre o desenvolvedor e usuário. Este tipo de situação ocorre porque o usuário tem dificuldades para expressar as suas necessidades.

5) O usuário tem em mente variáveis e atributos de alto nível (desempenho, confiabilidade, custos) já o desenvolvedor é totalmente ao contrário, sua mente fica mais associada a baixo nível, em todo o momento maquinando cálculos, situações que podem acontecer, em geral algoritmos.

> *A identificação das dificuldades e problemas pode servir como ponto inicial para as questões a ser discutidas durante a aplicação das técnicas de extração de requisitos. As solicitações ou requisições do usuário pode ser classificada de acordo com algumas características, que podem auxiliar no processo de extração de requisitos. São elas: a freqüência na requisição, a previsibilidade de solicitação e a atualização da informação. (CARVALHO e CHIOSSI, 2001, p.47)*

2.2.2. Técnicas para Extração de Requisitos

A extração de requisitos pode ser realizada de diversas formas. Uma das mais comuns é através de entrevistas preliminares.

O primeiro encontro com o desenvolvedor e o usuário, nada mais é do que um encontro de adolescentes, porque nenhum sabe bem o que dizer para o outro.

Segundo GAUSE e WEINBERG (1995), a comunicação deve ser iniciada fazendo perguntas de livre contexto, ou seja, um conjunto de perguntas que leve a uma concepção básica do problema, às pessoas que querem a solução, a natureza da solução que é desejada e a efetividade do primeiro encontro.

Um conjunto de perguntas de livre contexto que se pode levar em conta é:

- Quem esta por trás do pedido deste trabalho?
- Quem usará a solução?
- Qual o benefício econômico de uma solução bem sucedida?

- Qual problema esta solução resolverá?
- Como você caracteriza um bom resultado (saída) que seria gerado por uma solução bem sucedida?
- Você poderia mostrar-me o ambiente em que a solução será usada?

Segundo GAUSE e WEIBERG (1995), as perguntas concentram-se na efetividade do encontro:

> *- você é a pessoa certa para responder estas perguntas? suas respostas são oficiais?*
>
> *- minhas perguntas são pertinentes ao problema que você tem?*
>
> *- estou fazendo perguntas demais?*
>
> *- existe algo a mais que eu possa perguntar-lhe?*
>
> *- há mais alguém que possa fornecer informações adicionais?.*

As perguntas relacionadas acima são feitas para que a comunicação seja bem sucedida entre eles. Estas perguntas são feitas somente no início, as próximas abordariam soluções de problemas, negociações e especificações.

Para uma extração de requisitos de qualidade, é necessário que os dados sejam consistentes para obter um bom desempenho posteriormente. As pessoas são o principal meio para este processo, que deve ser bem organizado e bem distribuído, facilitando assim o entendimento do desenvolvedor do sistema.

As diversas formas de extração de requisitos se subdividem em formais e informais. As técnicas informais são baseadas na comunicação e interação com o usuário através de questionamentos, as formais pressupõe a construção de um modelo conceitual do problema sendo analisado, ou através de um protótipo do sistema. (CARVALHO e CHIOSSI, 2001, p.51)

2.2.2.1. Entrevistas

> *As entrevistas acontecem através de uma série de encontros, o usuário geralmente explica o seu trabalho, o ambiente no qual atua, e suas necessidades. Entretanto o entrevistador vai somente fazer perguntas; é uma técnica estruturada que pode ser aprendida e na qual os desenvolvedores podem ganhar proficiência com o treino e a prática. Requer o desenvolvimento de algumas habilidades sociais gerais, habilidade de ouvir e o conhecimento de variedades de táticas de entrevistas. As entrevistas constam em quatro fases: identificação do candidato, preparação, condução da entrevista e finalização. (CARVALHO e CHIOSSI, 2001, p.52)*

1) Identificação dos candidatos para a entrevista

A identificação das pessoas inicia desde o reconhecimento do problema de um determinado sistema. A busca de requisitos pode ser feita com várias pessoas, mas é importante saber se estas irão usar o sistema, pois as opiniões de usuários sobre o sistema serão muito mais pessoais. Exemplo "*Acho que ficaria melhor assim".*

2) Preparação para a entrevista

Para que uma entrevista seja de boa qualidade e tenha sucesso, é de extrema importância que o entrevistador prepare com antecedência a sua entrevista. É importante também que o entrevistador repasse para o entrevistado a duração e quais os objetivos que se deseja alcançar na entrevista. No início das entrevistas podem ser usados gravadores, porém este método pode deixar o entrevistado constrangido e levar a um mau desempenho em seus diálogos. As diversas perguntas a serem feitas vão surgindo com o decorrer da entrevista, não sendo possível elaborar todas previamente. (CARVALHO e CHIOSSI, 2001, p.52)

3) Condução da entrevista

> *O entrevistador faz uma breve revisão dos objetivos da entrevista, isto é, porque ela esta acontecendo, que destino terá a informação obtida, os tipos de assuntos que serão abordados, o tempo alocado para cada assunto. Durante esta revisão é possível julgar o quanto o entrevistado esta preparado; raramente a falta de preparo do entrevistado pode levar ao adiamento da entrevista. Apesar de a entrevista ser baseada em questões já preparadas, existem habilidades e estratégias para a comunicação oral que podem ser usadas para aumentar a qualidade da informação recebida. É preciso estar alerta para o fato de que a primeira resposta para a pergunta pode não estar necessariamente completa e correta; além disso, pode ser expressa em uma linguagem desconhecida para o entrevistador. Nesse caso, a melhor alternativa é resumir, refrasear e mostrar as aplicações do que o entrevistador esta ouvindo, de forma que o entrevistado possa confirmar o seu entendimento. (CARVALHO e CHIOSSI, 2001, p.53)*

4) Finalização da entrevista

Na finalização da entrevista é importante reservar uns 5 a 10 minutos para dar ênfase sobre os tópicos especiais, resaltando-os para o entrevistado.

2.2.2.2. *Brainstorming*

Consiste em várias reuniões onde as pessoas expõem suas idéias. Ela se divide em duas fases: a de geração de idéias e a de consolidação de idéias. "Na fase de geração, os participantes são encorajados a fornecer quantas idéias puderem, sem que haja discussão sobre o mérito delas. A técnica pode ser útil na geração de uma ampla variedade de visões do problema e formulação do problema de diferentes maneiras, na fase de consolidação as idéias são discutidas, revisadas e analisadas". (CARVALHO e CHIOSSI, 2001, p.55)

O *Brainstorming*, em geral, é uma forma útil na aquisição de requisitos de um sistema, pois a ausência de críticas e a quantidade de idéias geradas possibilitam a maior facilidade para a elaboração do produto final.

Para que o *Brainstorming* inicie, primeiramente há uma identificação geral dos participantes. O líder então agenda a sessão com todos em uma sala pré-estabelecida por ele. Após isto, o líder da sessão apresenta os problemas para os participantes, para que eles comecem a expor as suas sugestões. Devido às sugestões colocadas, é importante retirar as que mais se adequem ao sistema proposto. E então é feita a consolidação destas idéias, ou seja, a organização das mesmas, de maneira que possam ser melhor utilizadas.

2.2.2.3. *Pieces*

O *Pieces* é uma técnica usada na extração de requisitos. Esta técnica é usada na elaboração de questões de total necessidade para o sistema. A técnica de *Pieces* se limita a seis categorias para a elaboração de questões, como: *Desempenho, Informações e dados, Economia, Controle, Eficiência e Serviços.*

> *A técnica de Pieces ajuda a resolver esse problema fornecendo um conjunto de categorias de problemas que podem ajudar o analista a estruturar o processo de extração de*

> *requisitos. Pieces é uma sigla para seis categorias de questões a ser levadas em consideração: desempenho (ou performance), informações e dados, economia, controle, eficiência e serviços. Em cada categoria existem várias questões que o desenvolvedor deve explorar com os usuários. A técnica pode ser adaptada para incluir questões iniciais ou básicas que sejam especialmente relevantes para o tipo de produto a ser construído. A técnica de Pieces é mais proveitosa na análise de produtos de software já existentes, sejam manuais ou automatizados. (CARVALHO e CHIOSSI, 2001, p.57)*

2.2.2.4. Prototipagem

A prototipagem é um processo que possibilita ao desenvolvedor criar um modelo do sistema a ser implementado. O modelo de software pode ser iniciado de três formas: (1) feito em papel visando uma interação de homem e máquina; (2) a criação de subconjuntos de softwares, para a criação do sistema proposto; (3) avaliação de um sistema já existente.

A prototipagem inicia com a busca de requisitos, conforme figura 2.1. O desenvolvedor e o cliente juntam as idéias e as reúnem de forma a separar as que serão obrigatoriamente usadas. Isto leva a elaboração de um projeto rápido. "O projeto rápido concentra-se na representação daqueles aspectos do software que serão visíveis ao usuário (isto é, abordagens de entrada e formatos de saída). O projeto rápido leva a construção de um protótipo que é avaliado pelo cliente/usuário e é usado para refinar os requisitos para o software a ser desenvolvido".(PRESSMAN, 1995, p.36)

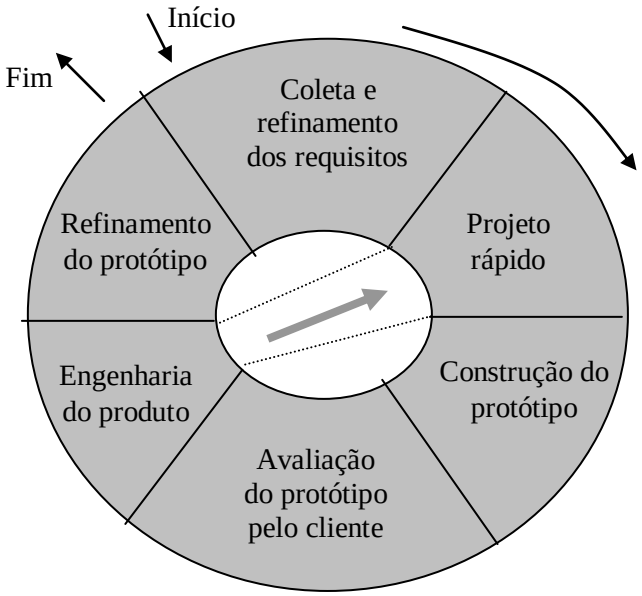


FIGURA 2.1 - Prototipação.
Fonte: (PRESSMAN, 1995, p.36)

2.3. Projeto (UML)

A UML (Unified Modeling Language) é uma das formas mais usadas para o desenvolvimento de sistemas em empresas. “O principal produto de uma equipe de desenvolvimento não são documentos bonitos, reuniões sofisticadas, ótimos slogans ou linhas de código merecedoras do prêmio *pulítzer* [1]. Última impressão em Criado em O principal produto é um bom software capaz de satisfazer as necessidades de seus usuários e respectivos negócios”. (BOOCH, RUMBAUGH e JACOBSON, 2000, p.03)

A modelagem é uma das partes principais das atividades que dão início a implantação de um bom software. Os modelos são usados nas mais diversas formas, visando a especificação e comunicação correta entre eles. “Os modelos ajudam a visualizar o sistema como ele é, ou como desejamos que seja. Os modelos permitem especificar a estrutura ou comportamento de um sistema. Os modelos proporcionam um guia para a construção do software. Os modelos documentam as tomadas de decisões”. (BOOCH, RUMBAUGH e JACOBSON, 2000, p.06)

2.3.1. A Linguagem UML

“A UML é uma linguagem padrão para a elaboração e construção de software. A linguagem pode ser usada para especificação, visualização, construção e documentação do sistema”. (BOOCH, RUMBAUGH e JACOBSON, 2000, p.13)

A linguagem UML se adequa para modelar sistemas tanto de informações, sistemas voltados a web e, até mesmo, sistemas complexos em tempo real. Para aprender e aplicar a UML em um sistema é necessário o entendimento de três regras principais; os blocos básicos de construção da UML; as regras de como esses blocos serão interligados; e os mecanismos que se aplicam a toda a linguagem.

A linguagem se destina a fatores de muita importância no momento da modelagem de um sistema. Ela permite, como visto acima, visualizar, especificar, construir e documentar os dados mais complexos de um software.

[1] O Pulitzer é patrocinado pela Universidade da Columbia e é oferecido a jornalistas, escritores e músicos que tenham feito algo significativo para a sociedade americana. O prêmio existe desde de 1917 e foi criado pelo empresário norte-americano Joseph Pulitzer.

2.3.1.1. Visualização

Para muitos programadores não existe a diferença em pensar em uma determinada implementação e passá-la para a codificação. Neste caso, o desenvolvedor está utilizando uma modelagem mental, no qual ele fará rabiscos em papéis. Este procedimento mental através de esboços pode acarretar em problemas, a menos que os desenvolvedores usem a mesma forma de demonstrar as suas anotações através de linguagens.

A UML é apresentada por símbolos gráficos concisos. Por trás de cada símbolo é observada a semântica de cada função. Desta forma, usando os padrões corretamente, qualquer desenvolvedor poderá interpretá-lo sem nenhuma dificuldade.

2.3.1.2. Especificação

A especificação nos métodos da UML significa construir modelos corretos, sem falhas e sem ambigüidades. “Em particular, a UML atende a todas as decisões importantes em termos de análise, projeto e implementação, que devem ser tomadas para o desenvolvimento e implantação de sistemas complexos de softwares”.(BOOCH, RUMBAUGH e JACOBSON, 2000, p.15)

2.3.1.3. Construção

A UML não é uma linguagem visual, mas os modelos UML podem ser adaptados às diversas linguagens de programação.

2.3.1.4. Documentação

A UML se torna uma linguagem de documentação devido a alguns artefatos como: requisitos, arquitetura, projeto, código-fonte, planos-do-projeto, testes, protótipos e versões.

Dependendo de como está o desenvolvimento do sistema, alguma destas características não é parte a ser entregue ao usuário, elas são usadas para controlar as manutenções após a implantação.

2.3.2. Diagramas da UML

Um diagrama UML é composto basicamente de elementos gráficos, ou seja, desenhos. Estes desenhos são feitos para uma melhor visualização dos diversos aspectos de um sistema.

Segundo BOOCH, RUMBAUGH e JACOBSON (2000), a UML fornece os seguintes diagramas:

- Diagrama de Classes: Um diagrama de classes mostra um conjunto de classes. São geralmente usados e encontrados em modelagens orientadas a objetos, onde abrangem uma visão estática do sistema.

- Diagrama de Objetos: Um diagrama de objetos mostra seus objetos e relacionamentos. Traz as instâncias que são encontradas no diagrama de classes, ou seja, as variáveis. Também mostram uma visão estática de um processo do sistema.

- Diagrama de Casos de uso: O diagrama de casos de uso usa um tipo especial de classe, os atores que representam os usuários. Estes diagramas são importantes porque apresentam como será a interação entre o usuário e o sistema.

- Diagrama de Seqüência: Este diagrama pode ser visto como um diagrama de interação. Exibe interação dos objetos e seus respectivos relacionamentos. Além disso, inclui as mensagens trocadas entre os objetos. Este diagrama mostra uma visão dinâmica do sistema.

- Diagrama de Colaboração: O diagrama de colaboração é um diagrama de interação. Ele se preocupa em organizar as mensagens trocadas entre os objetos, tanto os que enviam quanto os que recebem.

- Diagrama de Gráficos de Estado: Estes diagramas abrangem uma visão dinâmica

de um sistema. Preocupa-se em exibir transições, eventos e atividades do sistema. São importantes porque analisam os comportamentos que acontecem nas interfaces.

- Diagrama de Atividades: O diagrama de atividades basicamente mostra as atividades do sistema e como os fluxos ocorrem de uma atividade para a outra. Mostra uma visão dinâmica do sistema e dá ênfase no controle de fluxos dos objetos.

- Diagrama de Componentes: Um diagrama de componentes abrange uma visão estática do sistema e se preocupa em organizar as dependências existentes em um conjunto de componentes.

- Diagrama de Implantação: O diagrama de implantação mostra a configuração em tempo de execução dos componentes que já existem nele. Está ligado ao diagrama de componentes, porque um nó inclui um ou mais componentes (um para muitos). O diagrama de implantação tráz uma visão estática do seu funcionamento.

2.3.3. Diagrama de Robustez

Além dos diagramas pertencentes a UML, neste trabalho foi utilizado o Diagrama de Robustez. Este diagrama foi criado para ser um intermediário entre o Diagrama de Casos de Uso e o Diagrama de Seqüência. Nele são mostrados os objetos do sistema e suas interações, ou seja, possui componentes que representam os atores (usuários) e componentes de interface (formulários do sistema), controle (controle das mensagens) e armazenamento (Banco de Dados).

Este Diagrama é de extrema importância, pois é a base para a criação do Diagrama de Seqüência.

2.4. Implementação e Testes

A partir das definições gerais do programa, o programador dá início às tarefas de passar para uma linguagem de programação escolhida por ele, ou às vezes escolhida pelo próprio usuário, do sistema previamente modelado.

Na hora da implementação, principalmente se for um sistema de grande porte, é importante que a fase de testes seja feita a cada módulo, ou seja, a cada função, para que mais adiante os testes entre as suas interligações sejam menores.

O teste piloto, pouco conhecido, ou melhor, pouco praticável é um bom método para colocar em prática, porque consiste em simular o seu funcionamento sabendo as condições reais de suas operações. Quando é localizado um erro no sistema é importante retornar para o teste piloto acompanhado pelos usuários principais.

Após terminados os testes do sistema, é de total interesse da empresa fazer treinamento junto ao desenvolvedor para os possíveis usuários, dando ênfase na segurança dos dados.

É importante que o sistema tenha uma manual do sistema, o qual será utilizado no departamento de processamento de dados. Este manual deve explicar todo o seu funcionamento, desde procedimentos operacionais até a parte de interfaces visando o fácil manuseio e entendimento do sistema.

3. PROJETO DE BANCO DE DADOS

3.1. Introdução

Um Sistema Gerenciador de Banco de Dados (SGBD) é constituído por um conjunto de programas, através dos quais permite acesso aos dados armazenados. Estes conjuntos de dados são chamados de Banco de Dados, que contém informações sobre um determinado sistema.

O principal objetivo de usar um Banco de Dados é possuir um ambiente seguro e eficiente para o armazenamento e recuperação dos dados.

3.2. Vantagens de um Sistema Gerenciador de Banco de Dados

Segundo SILBERSCHATZ, KORTH e SUDARSHAN (1999), a utilização de um Banco de Dados apresenta algumas vantagens em relação ao armazenamento em arquivos isolados. As principais são:

- Inconsistência e redundância de dados: antes do surgimento do SGBD´s, os dados das aplicações eram armazenados em arquivos separados. Assim, uma informação poderia ser armazenada em mais de um local, levando a redundância. Este problema gerava inconsistência, pois em algumas situações a aplicação atualizava um dado em um arquivo e não o atualizava em outro, armazenando assim dois valores diferentes para um mesmo dado.

- Dificuldade de acesso aos dados: como os dados estavam espelhados em diferentes arquivos, muitas vezes tornava-se difícil obter alguma informação que necessita-se

da união de dados de diferentes arquivos.

- Problemas de integridade: nos sistemas antigos era difícil garantir a integridade dos dados após alguma atualização, pois não era fácil determinar exatamente os reflexos que ela poderia trazer em outros arquivos.

- Problemas de atomicidade: um SGBD garante a atomicidade das transações.

> *Um sistema computacional, como qualquer outro dispositivo mecânico ou elétrico, está sujeito à falhas. Em muitas aplicações é crucial assegurar que, uma vez detectada uma falha, os dados sejam salvos em seu último estado consistente, anterior a ela. Considere um programa para transferir 50 dólares da conta A para uma conta B. Se ocorrer falha no sistema durante sua execução, é possível que os 50 dólares sejam debitados da conta A sem serem crediados na conta B, criando um estado inconsistente no Banco de Dados. Logicamente, é essencial para a consistência do Banco de Dados que ambos, debito e créditos, ocorram ou nenhum deles seja efetuado. Isto é, a transferência de fundos deve ser uma operação atômica – deve ocorrer por completo ou não ocorrer. É difícil garantir essa propriedade em um sistema convencional de processamento de arquivos. (SILBERSCHATZ, KORTH e SUDARSHAN, 1999, p.03)*

- Problemas de segurança: o armazenamento dos dados em arquivos isolados dificulta a segurança dos mesmos, permitindo que pessoas acessem dados que não dizem respeito do seu trabalho. Os SGBD´s possuem mecanismos para controlar a permissão de acessos de usuários específicos ou grupos de usuários.

Um Banco de Dado é projetado para guardar um volume expressivo de informações, por isso, é importante que seja bem projetado de modo a evitar a ocorrência de erros futuros.

Todo projeto de um sistema que armazena dados necessita de uma modelagem. Esta é feita através de modelos de Entidade-Relacionamento.

> *O objetivo da Modelagem de Dados é transmitir e apresentar uma representação única, não redundante e resumida, dos dados de uma aplicação. Em projetos conceituais de aplicação em Banco de Dados o modelo Entidade-Relacionamento é o mais largamente utilizado para a representação e entendimento dos dados que compõem e essência do sistema. O projeto de Banco de Dados para sistemas de aplicação hoje não é mais uma tarefa realizada somente por profissionais da área de informática, mas também possível de ser realizada por não especialistas, através de técnicas estruturadas como a Modelagem Conceitual de Dados. O projeto de um sistema de informação é uma atividade complexa que inclui planejamento, especificações e desenvolvimento de vários componentes. (MACHADO e ABREU, 1996, p.30)*

3.3. Abstração dos Dados

Um sistema de gerenciamento de Banco de Dados (SGBD), permite que os dados sejam visualizados em diferentes níveis de abstração, dependendo de quem é a pessoa interessada nas informações.

> *Para que se possa usar um sistema, ele precisa ser eficiente na recuperação das informações. Esta eficiência está relacionada à forma pela qual foram projetadas as complexas estruturas de representação desses dados no Banco de Dados. Já que muitos dos usuários dos sistemas de Banco de Dados não são treinados em computação, os técnicos em desenvolvimento de sistemas omitem essa complexidade desses usuários por meio dos diversos níveis de abstração, de modo a facilitar a interação dos usuários com o sistema. (SILBERSCHATZ, KORTH e SUDARSHAN, 1999, p.04)*

- Nível Físico: este nível tem a função de mostrar, como os dados estão armazenados no Banco de Dados. Ou seja, como um registro é armazenado, não sendo interessante do ponto de vista da aplicação em si.

- Nível Lógico: neste nível, a principal função é de mostrar quais dados estão armazenados no Bando de Dados, e as suas interligações. No nível lógico a preocupação de abstração dos dados é necessariamente elaborado pelo administrador de Banco de Dados, isto é, ele que realmente precisa decidir quais dados devem pertencer ao Banco de Dados.

- Nível de Visão: É o nível mais alto entre os três níveis de abstração. Ele mostra apenas parte do Banco de Dados. O nível de visão é definido para que os usuários finais não precisem saber todas informações contidas na base, e sim apenas as informações relevantes. Diferentes usuários podem acessar diferentes visões, dependendo das informações que lhes interessam, conforme a figura 3.1.

3.4. Pessoas envolvidas em um Banco de Dados

3.4.1. O Administrador

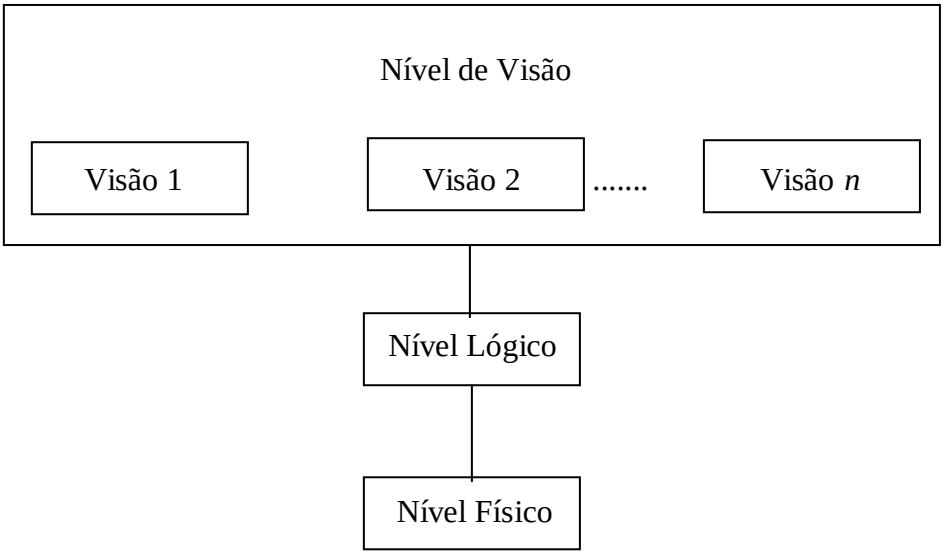


FIGURA 3.1 - Os três níveis de abstração de dados.
Fonte: (SILBERSCHATZ, KORTH, SUDARSHAN, 1999, p.5)

As principais funções de um Administrador de Banco de Dados são as seguintes:

- A definição do esquema do Banco de Dados, onde é descrito a estrutura das tabelas, com seus campos, chaves, relacionamentos etc.
- A autorização de acesso ao sistema. O Administrador faz o controle de acesso aos dados entre ou usuários, especialmente que pessoas ou conjunto de pessoas podem acessar os dados ou parte deles e o que eles podem fazer com estes dados. (consultas, atualizações, inclusões, exclusões).

3.4.2. Usuários Finais

Os usuários de um Banco de Dados se diferenciam em vários grupos como: os programadores de aplicação, os usuários sofisticados, os usuários especializados e por fim os usuários navegantes.

Os programadores de aplicação são os programadores que interagem com o sistema existente. A sua função é adequar novas aplicações para um sistema de conhecimento dele, usando linguagens de programação. Os usuários sofisticados interagem com o sistema, mas não ajudam na sua escrita. Estes usuários colaboram com a formulação de consultas ou outras operações através de linguagens adequadas.

Os usuários especialistas são pessoas que escrevem aplicações especializadas para um Banco de Dados. Aplicações que não são definidas como tradicionais para um processamento de dados. Os usuários navegantes são os que interagem com um sistema qualquer, sem a participação em escritas e aplicações como as outras.

3.5. Diagramas Entidade-Relacionamento

O Modelo Entidade-Relacionamento é formado por um conjunto de símbolos que representam entidades, atributos e relacionamentos. Este modelo foi desenvolvido para facilitar o projeto de um Banco de Dados e ilustrar os dados armazenados no sistema e os relacionamentos entre eles. Os símbolos envolvidos em um DER são:

3.5.1. Entidades e Atributos

As entidades desempenham um papel importante em um diagrama de Entidade-Relacionamento. “Define-se entidade como aquele objeto que existe no mundo real como uma identificação distinta e com um significado próprio, são as “coisas” que existem no negócio, ou ainda, descrevem o negócio em si”. (MACHADO e ABREU, 1996, p.34)

Nas entidades é que estão os atributos, ou seja, as variáveis do sistema.

Os atributos são, por exemplo, o nome de um cliente entre outros como mostra a figura 3.2.

Os atributos que descrevem uma entidade possuem valores diferenciados para o seu armazenamento.

“Segundo SILBERSCHATZ, KORTH e SUDARSHAN (1999), os atributos são classificados em”:

- Atributo Simples: um atributo simples é aquele que não se divide, por exemplo, o RG de uma determinada pessoa.

- Atributo Composto: um atributo composto é aquele que se divide em partes.

Entidade Pessoa

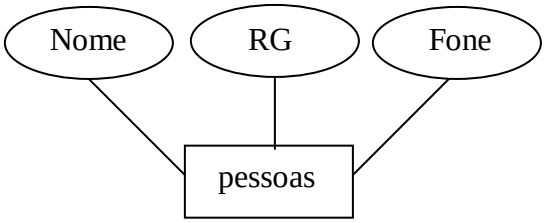


FIGURA 3.2 - Conjunto da Entidade Pessoa e seus Atributos.

Como por exemplo, tem-se o endereço de uma pessoa, onde a divisão deste atributo abrangeria a rua, telefone, CEP, bairro, entre outras.

- Atributo Monovalorado: um atributo monovalorado é aquele que armazena um único valor por vez.

- Atributo Multivalorado: um atributo multivalorado é o contrário do monovalorado, ou seja, pode aceitar um conjunto de valores.

- Atributo Nulo: um atributo nulo é aquele sem valor. Não possui nenhum tipo de informação referente ele.

- Atributo Derivado: um atributo derivado é aquele que pode ser calculado a partir de outras entidades a ele relacionado.

3.5.2. Relacionamento

Os relacionamentos são descritos para mostrar quais entidades estão ligadas. Os relacionamentos são importantes, por facilitarem a observação das ligações em um diagrama de Entidade-Relacionamento.

Um relacionamento deve representar a cardinalidade, ou seja, a quantidade de instâncias que podem estar envolvidas naquela relação. As cardinalidades são: um para um, um para muitos, muitos para um, muitos para muitos.

O diagrama mostrado na figura 3.3 é compostos de duas entidades (pessoas e apartamentos), um relacionamento (alugam) e a sua cardinalidade, que representa a quantidade de apartamentos que uma pessoa pode ter, ou vice-versa, a quantidade de pessoas que podem locar um apartamento.

Neste caso, o relacionamento é de muitos para muitos, como mostra a figura 3.3. Uma instância da entidade pessoas é associada com várias instâncias da entidade apartamento, e vice-versa, ou seja, 1 pessoa pode ter vários apartamentos e um apartamento pode ser locado por várias pessoas.

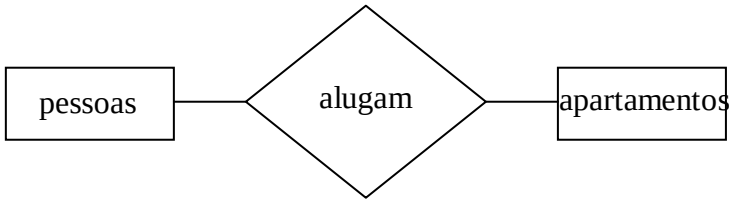


FIGURA 3.3 - Relacionamento Muitos para Muitos.

3.5.3. Especialização e Generalização

A especialização em um modelo de Entidade-Relacionamento significa criar uma entidade em um diagrama que agrupe características (atributos comuns) e em seguida, derivar novas classes a partir desta, as quais conterão além dos atributos comuns (herdados da classe de origem) atributos específicos a cada uma das novas classes. Esse método é chamado de Top-Down.

> *A especificação parte de um único conjunto por meio de entidades; ela enfatiza as diferenças entre as entidades pertencentes ao conjunto por meio do estabelecimento das diferenças expressas nos conjuntos de entidades de nível inferior. Estes conjuntos de entidades de nível inferior podem possuir atributos, ou mesmo participar de relacionamentos que não podem ser aplicados a todas as entidades do conjunto de entidades de nível superior. (SILBERSCHATZ, KORTH e SUDARSHAN, 1999, p.42)*

A generalização no modelo de Entidade-Relacionamento consiste em iniciar o processo dos níveis inferiores. É chamado de método Button-Up, parte de classes específicas e vai generalizando em entidades, até chegar em uma classe que abranja as demais. Este método é simplesmente o inverso da especialização. A figura 3.4 ilustra a especialização e generalização através do símbolo ISA.

O uso da generalização procede para o reconhecimento de um número de conjuntos de entidades que compartilham características comuns (são descritos pelos mesmos atributos e participam dos mesmos conjuntos de relacionamentos). Com base nessas características comuns, a generalização sintetiza esses conjuntos de entidades em um só conjunto de entidades de nível superior. A generalização é usada para enfatizar as similaridades entre os conjuntos de entidades de nível inferior. Omitindo as suas diferenças; isso permite também uma representação mais econômica, evitando repetições de atributos compartilhados. (SILBERSCHATZ, KORTH e SUDARSHAN, 1999, p.42)

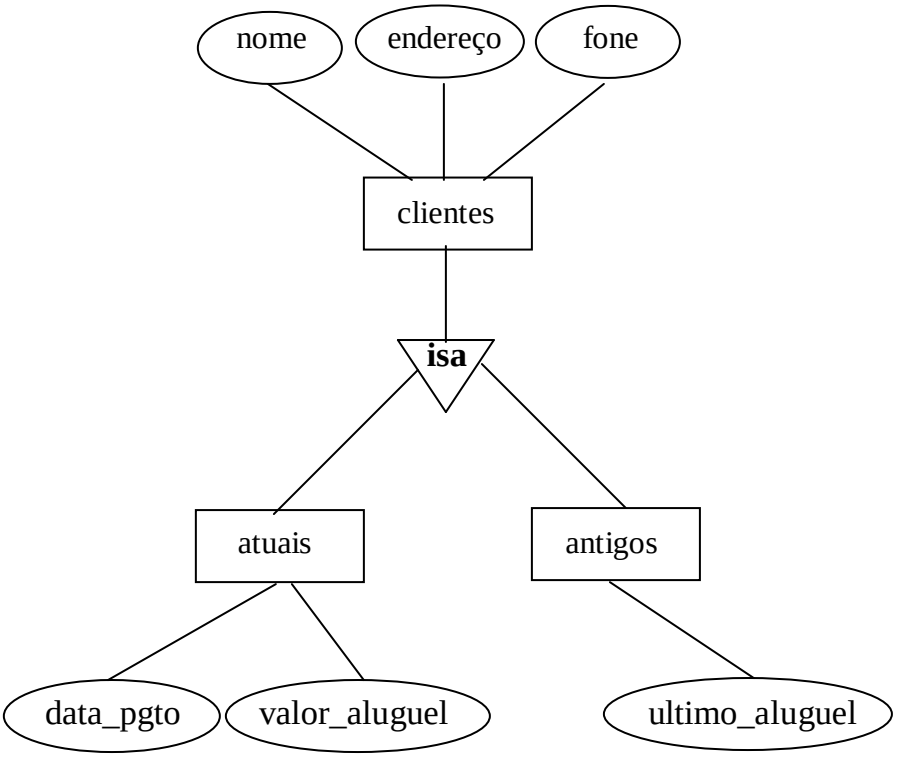


FIGURA 3.4 - Generalização e Especialização.
Fonte: (SILBERSCHATZ0, KORTH, SUDARSHAN, 1999, p.41)

3.5.4. Herança de Atributos

A herança de atributos ocorre na especialização e na generalização. As entidades de nível inferior herdam atributos das entidades de nível superior a elas relacionadas. Deste modo à declaração dos atributos é feita uma única vez.

Se uma dada porção do modelo E-R chegou à especialização ou à generalização, os resultados são basicamente os mesmos:

- Conjunto de entidades de nível superior com atributos e relacionamentos que são aplicados a todos os seus conjuntos de entidades de nível inferior.

- Conjuntos de entidades de nível inferior com características distintas que são apenas aplicadas a um conjunto de entidades de nível inferior em particular. (SILBERSCHATZ, KORTH e SUDARSHAN, 1999, p.42)

A figura 3.4 ilustra claramente a herança de atributos, isto é, a entidade cliente possui os atributos comuns às entidades atuais e antigos. Desta forma as entidades de nível inferior (atuais e antigos) possuem os seus atributos específicos além dos atributos herdados da entidade de nível superior (cliente).

3.6. Considerações Finais

Um Banco de Dados consistente é de fundamental importância para o sucesso de um projeto. Muitas vezes um projeto mal estruturado de um Banco de Dados, causa problemas em sua consistência e fica com um grau de confiabilidade baixo causando muitas falhas. Por isso, no projeto de um Banco de Dados os diagramas devem ser bem definidos para que o projeto final tenha sucesso.

4. MODELAGEM DO SISTEMA

4.1. Introdução

Neste capítulo, será apresentada a modelagem do sistema. Inicialmente é feita a especificação dos requisitos usando os diagramas de use-case.

Com o diagrama de use-case feito, são descritos os cenários para cada uma das funções. Nos cenários é apresentado textualmente as etapas para que a operação seja efetuada.

Em seguida é dado início ao projeto de software. Para isto foram escolhidos os diagramas de robustez e de seqüência onde mostram os objetos envolvidos e as mensagens trocadas entre eles.

4.2. Domínio do Problema

O trabalho versará sobre a elaboração de um sistema para gerenciar uma pizzaria.

O sistema controlará os produtos e os pedidos efetuados. Além de gerar a conta da mesa, o sistema será responsável por fazer o controle de estoque dos ingredientes.

O sistema deve permitir a inclusão, exclusão, alteração de produtos e ingredientes, além de fornecer relatórios para um controle de estoque mais eficiente.

Estas informações foram obtidas através de técnicas de obtenção de requisitos. Para isto, foram realizadas entrevistas e estudou-se o sistema atual para identificar os requisitos do sistema e seu funcionamento.

4.3. Especificação de requisitos e Análise de Robustez

4.3.1. Diagramas de Use-Case e Robustez

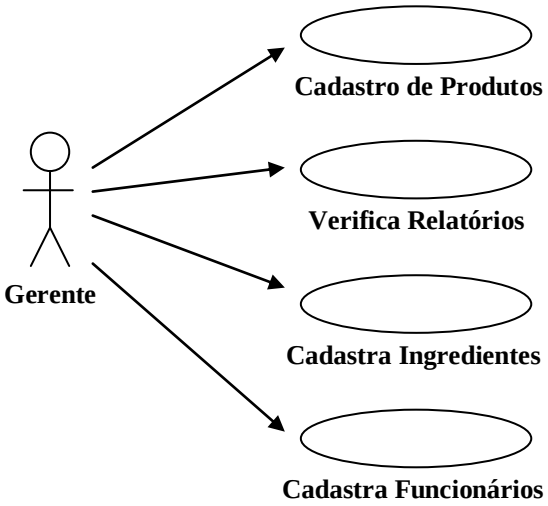
Os diagramas de use-case são utilizados para representar os requisitos de um sistema. Seus componentes são os atores que representam os usuários; os casos de uso, representados por elípses, que indicam as funções que o ator pode executar no sistema; e as ligações que indicam as associações existentes entre os atores e os casos de uso, que são representadas por setas.

O diagrama de robustez é importante porque mostra para o usuário do sistema as principais funções atavés de desenhos e setas mostrando as suas interligações. Ele serve como base para o desenvolvimento do diagrama de sequência, pois apresenta os objetos envolvidos.

Estes diagramas são de extrema importância porque apresentam como será a interação entre o usuário e o sistema

4.3.2. Especificação do Sistema

A figura 4.1. apresenta o diagrama de Use-case para o sistema proposto.



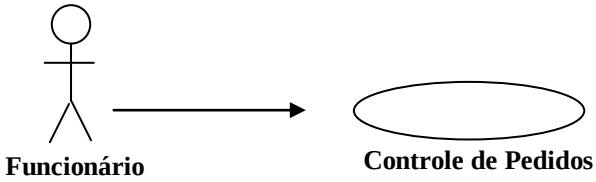


FIGURA 4.1 – Diagrama de Use-case.

4.3.3. Descrição de Cenários

Para cada requisito do sistema, é feita uma descrição textual do seu funcionamento, abordando as suas principais funções.

Cadastro de Produtos

Nome: Cadastro de Produtos.
Objetivo: Controlar os produtos em estoque.
Inicia: Inicia com o gerente escolhendo a opção controle de produtos.
Fluxo Principal:

- O sistema fornece uma tela contendo os dados necessários para que a operação seja efetuada no Banco de Dados.
- O gerente digita as informações necessárias.
- O gerente confirma a operação.

Fluxo Alternativo:

- Dados insuficientes para efetuar a operação.
- O gerente a qualquer momento pode sair do sistema.

Termino: Após feita a operação, o sistema retorna para a tela principal.

A Figura 4.2 apresenta o diagrama de robustez para o use-case cadastro de produtos, nela pode-se observar os objetos como a “form principal”, que através de um botão chama o “formulário de cadastro” e em seguida um botão para “confirmar”.

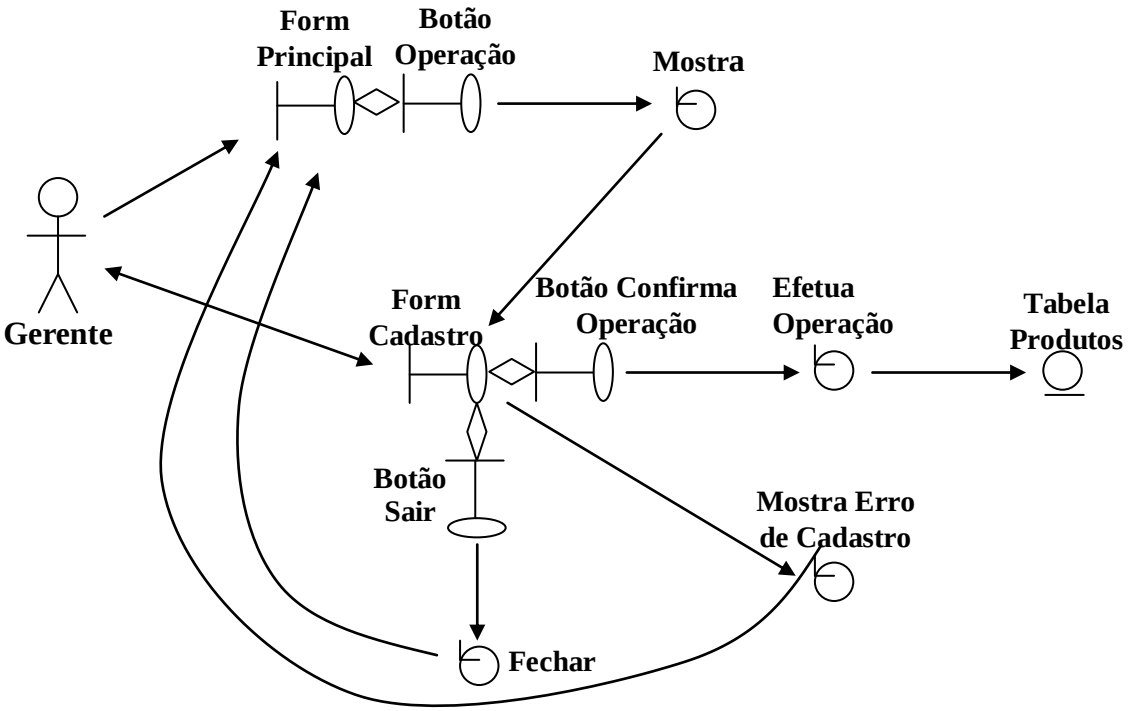


FIGURA 4.2 – Diagrama de Robustez - cadastro de produtos

Verifica Relatórios

Nome: Verifica Relatórios.
Objetivo: O objetivo é gerar relatórios da situação do estoque e dos pedidos realizados.
Inicia: Inicia com o gerente escolhendo a opção gerar relatórios.
Fluxo Principal:

- O gerente escolhe o tipo de relatório. (Pedidos ou Estoque)
- Os dados são digitados pelo gerente de acordo com a sua escolha.
- O gerente decide se efetua ou não a impressão do relatório.

Término: Após a confirmação ou não-confirmação da impressão, o sistema retorna para a tela principal.

A Figura 4.3 apresenta o diagrama de robustez para o use-case verifica relatório, nele pode-se observar os objetos como a “form principal”, que através de um botão mostra o “tipo de relatório” e em seguida um botão para “gerar o relatório” escolhido.

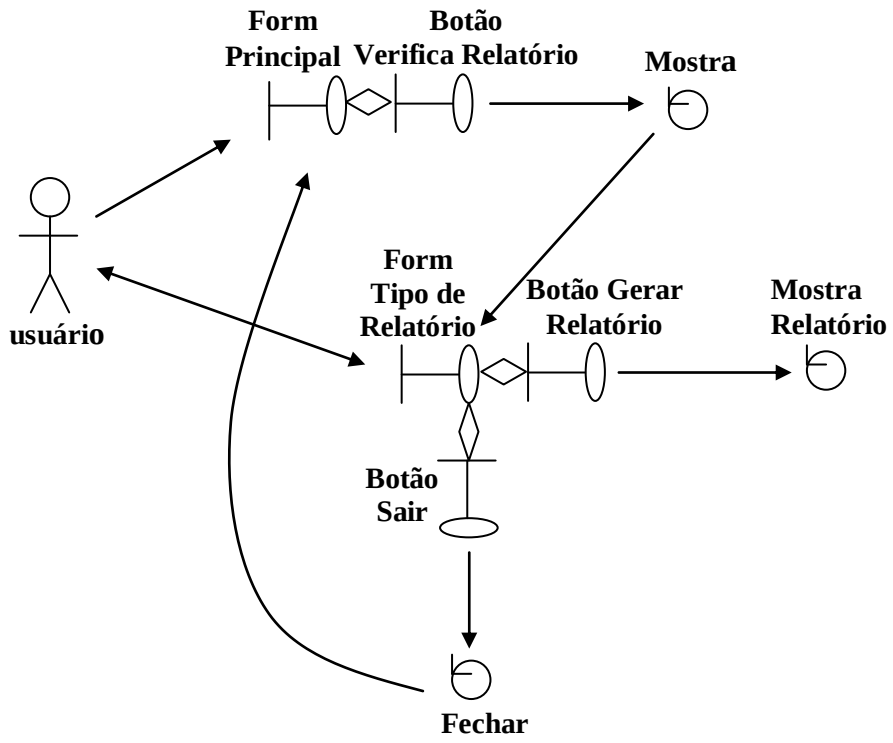


FIGURA 4.3 – Diagrama de Robustez - verifica relatório

Cadastro de Ingredientes

Nome: Cadastro de Ingredientes.

Objetivo: Controlar os ingredientes em estoque.

Inicia: Inicia quando o cliente escolhe a opção controle ingredientes

Fluxo Principal:

- O sistema fornece uma tela contendo os dados necessários para que a operação seje efetuada no Banco de Dados.
- O gerente digita as informações necessárias.
- O gerente confirma a operação.

Fluxo Alternativo:

- Dados insufucientes.
- O usuário a qualquer momento pode sair do sistema.

Término: Após confirmar da operação, o sistema retorna para a tela principal.

A Figura 4.4 apresenta o diagrama de robustez para o use-case cadastro de ingredientes, nele pode-se observar os objetos como a “form principal”, que através de um botão chama o “formulário de cadastro” e em seguida um botão para “confirmar” e “efetuar a operação” no Banco de Dados referente à “tabela ingredientes”.

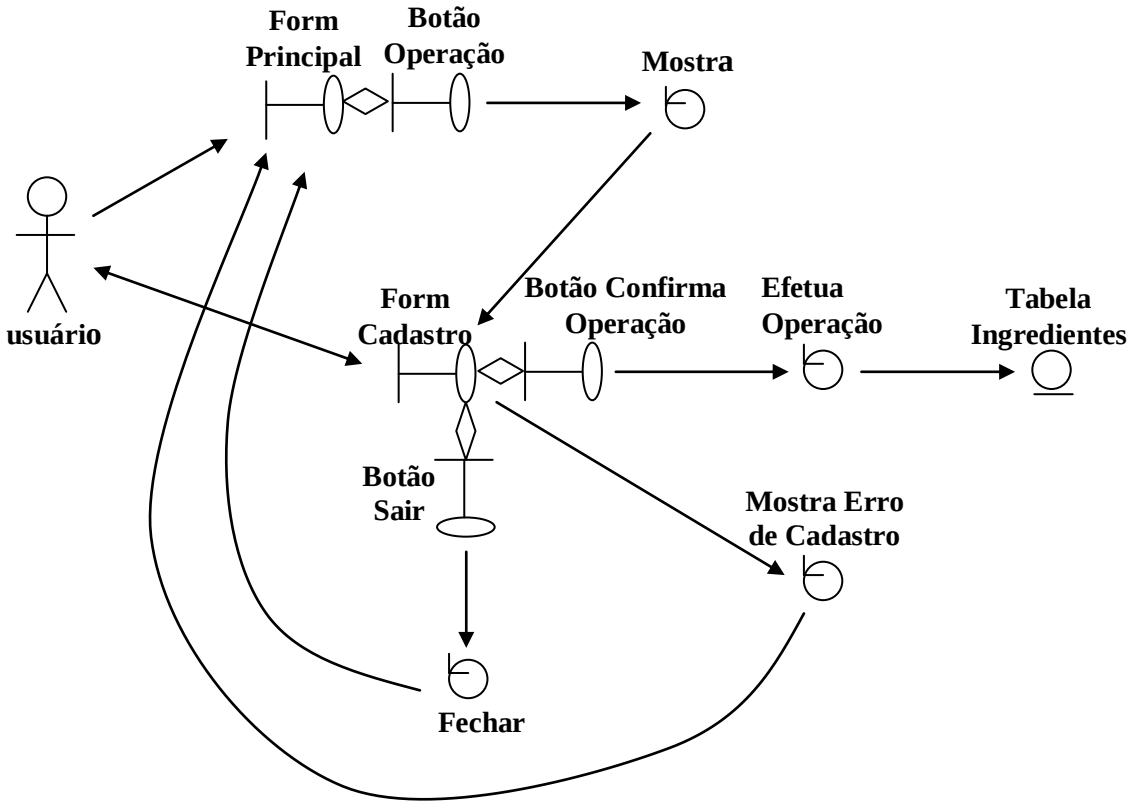


FIGURA 4.4 – Diagrama de Robustez - cadastro de ingredientes

Cadastro de Funcionários

Nome: Cadastro de Funcionários.
Objetivo: O objetivo é de fazer um controle dos funcionários.
Inicia: Inicia com o usuário escolhendo a opção cadastro de funcionários.
Fluxo Principal:

- O sistema fornece uma tela contendo os dados necessários para que a operação seja efetuada no Banco de Dados.
- O gerente entra com os dados referentes aos funcionários.
- O gerente confirma a operação.

Fluxo Alternativo:

- Dados insuficientes.
- O usuário a qualquer momento pode sair, e cancelar a operação.

Término: Após a operação efetuada, o sistema retorna para a tela principal.

A Figura 4.5 apresenta o diagrama de robustez para o use-case cadastro de funcionários, nela pode-se observar os objetos como a "form principal", que através de um botão chama o "formulário de cadastro" e em seguida um botão para "confirmar" e "efetuar a operação" no Banco de Dados referente à "tabela funcionários".

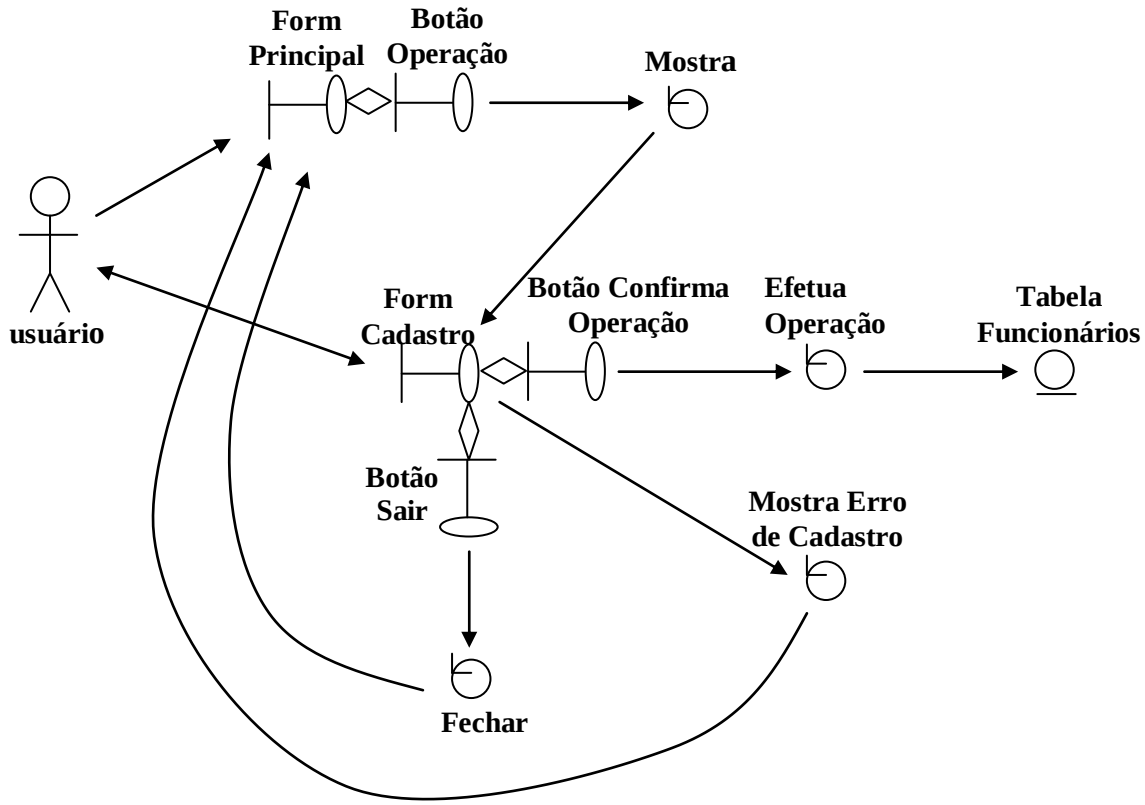


FIGURA 4.5 – Diagrama de Robustez - cadastro de funcionários

Controle de Pedidos

Nome: Controle de Pedidos.
Objetivo: Controlar os pedidos feito pelos clientes.
Inicia: Inicia com o usuário escolhendo a opção controle de pedidos.
Fluxo Principal:

- O sistema fornece uma tela contendo os dados necessários para que a operação seja efetuada.
- O funcionário digita os dados referente a escolha do pedido.
- O funcionario confirma a operação e o sistema dá baixa no estoque.

Fluxo Alternativo:

- Dados insuficientes ou incorretos para completar a operação.
- O usuário a qualquer momento pode sair, e cancelar a operação.

Término: Após a efetuada a operação, o sistema retorna para a tela principal.

A Figura 4.6 apresenta o diagrama de robustez para o use-case controle de pedidos, nela pode-se observar os objetos como a “form principal”, que através de um botão chama o “formulário de Controle de pedidos” e em seguida um botão para “executar a operação” no Banco de Dados referente à “tabela de pedidos”.

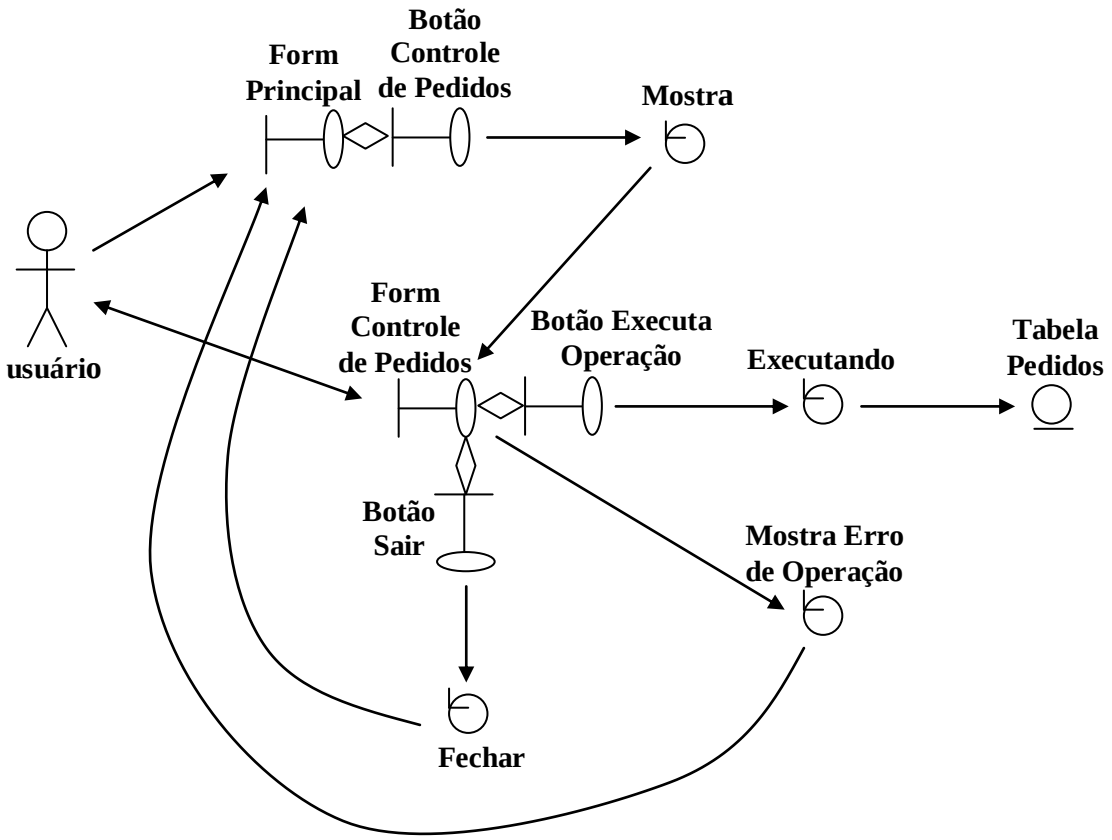


FIGURA 4.6 – Diagrama de Robustez - controle de pedidos

4.4. Diagrama de Seqüência

Esta parte é a fase onde se inicia o projeto de software. Para cada requisito funcional do software a ser construído, é elaborado um diagrama que indica quais mensagens serão trocadas entre os objetos.

Na figura 4.7 é descrito todo o processo para que o produto seja cadastrado, alterado ou excluído do Banco de Dados. Estas indicações são representadas por setas, ou seja, cada uma focando diretamente sobre os objetos principais do formulário. O gerente efetua uma chamada no botão de operação para fazer a chamada do cadastro e depois de escrito os valores ele verifica os dados e confirma a operação na tabela de produtos.

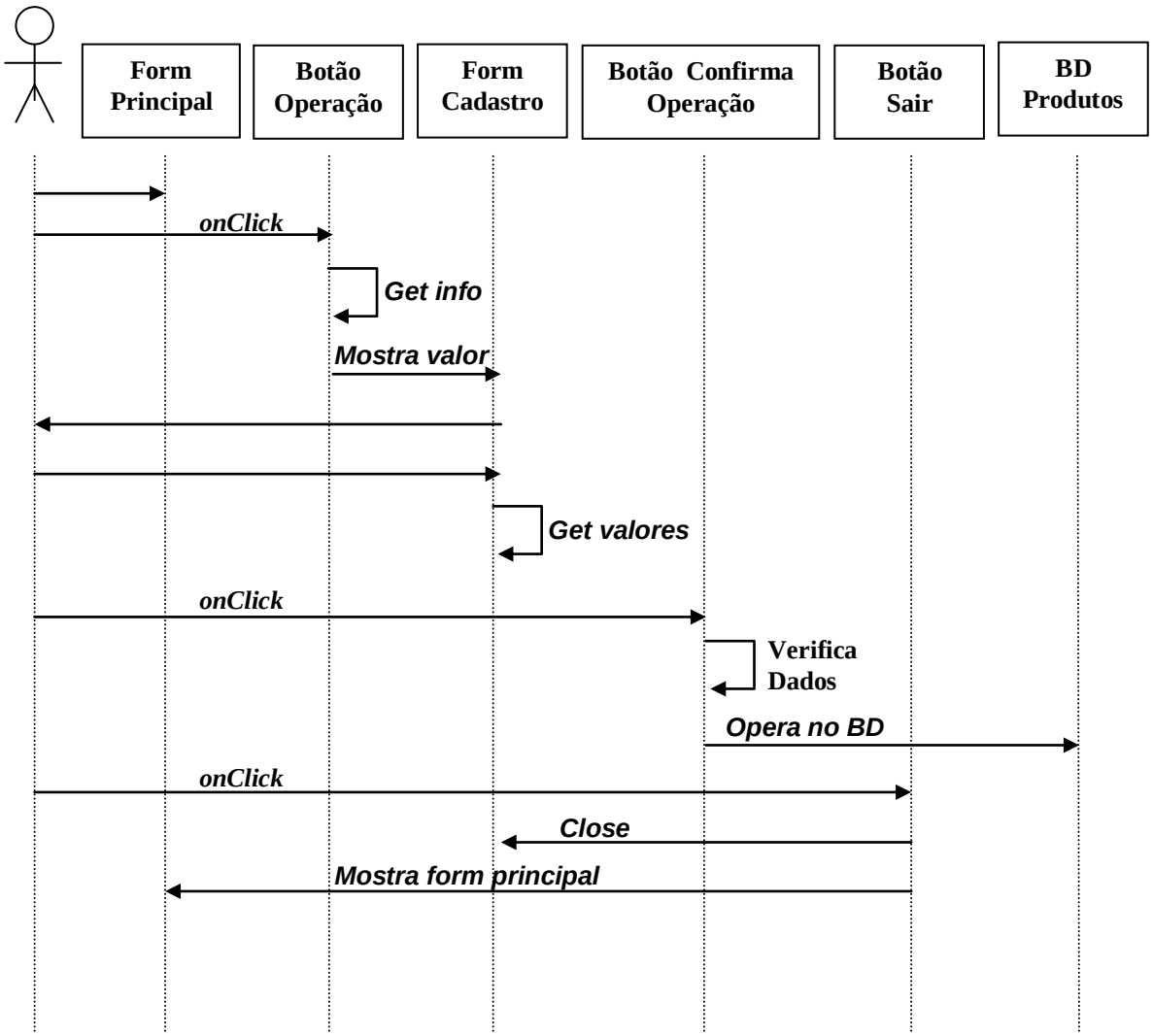


FIGURA 4.7 – Diagrama de Seqüência - cadastro de produtos

Na figura 4.8 é descrito todo o processo para que seja criado um relatório de acordo com a sua escolha. O gerente efetua uma chamada no botão verifica relatório para fazer a chamada do tipo de relatório e depois de escolhido ele gera os valores e confirma a operação para gerar o relatório.

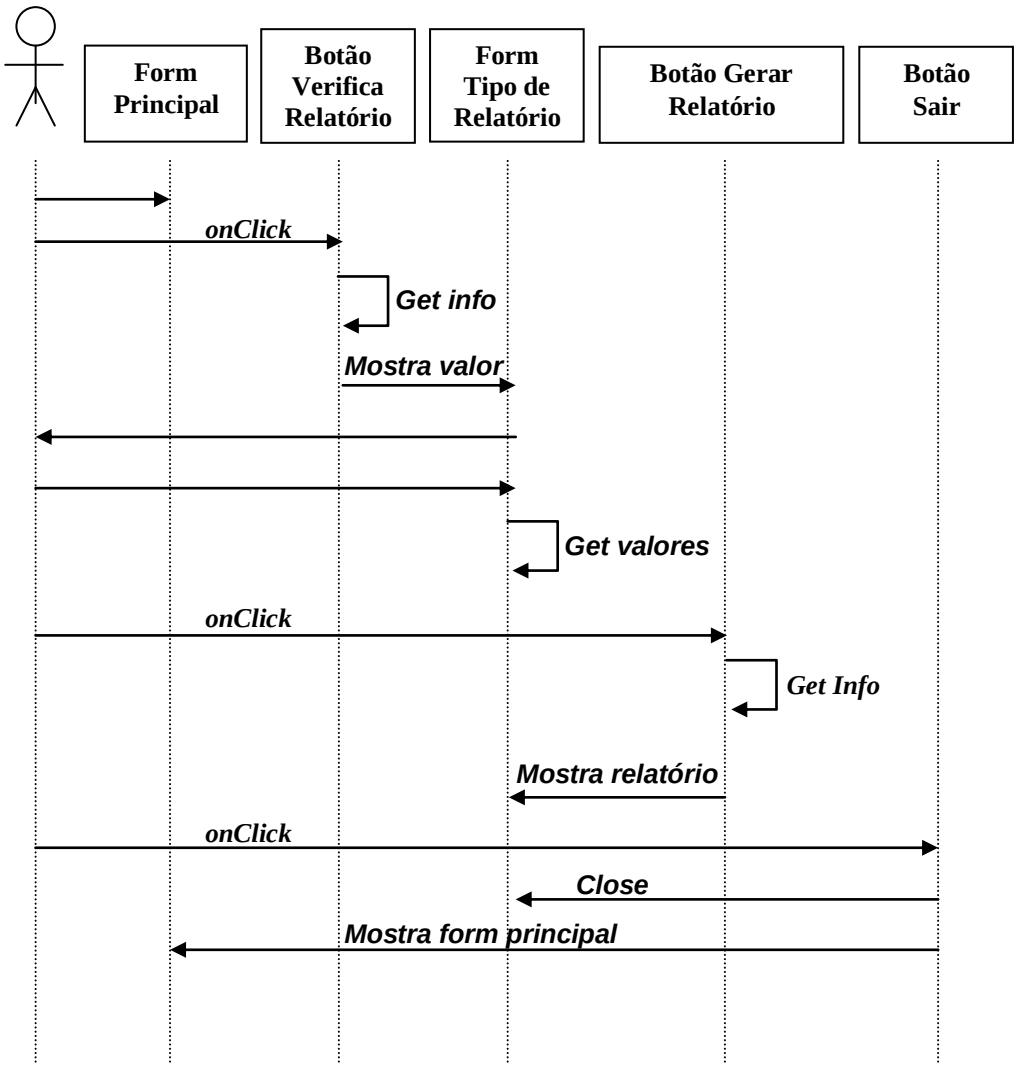


FIGURA 4.8 – Diagrama de Seqüência - verifica relatório

Na figura 4.9 é descrito todo o processo para que o ingrediente seja cadastrado, alterado ou excluído do Banco de Dados. O gerente efetua uma chamada no botão de operação para fazer a chamada do cadastro e depois de escrito os valores ele verifica os dados e confirma a operação na tabela de ingredientes.

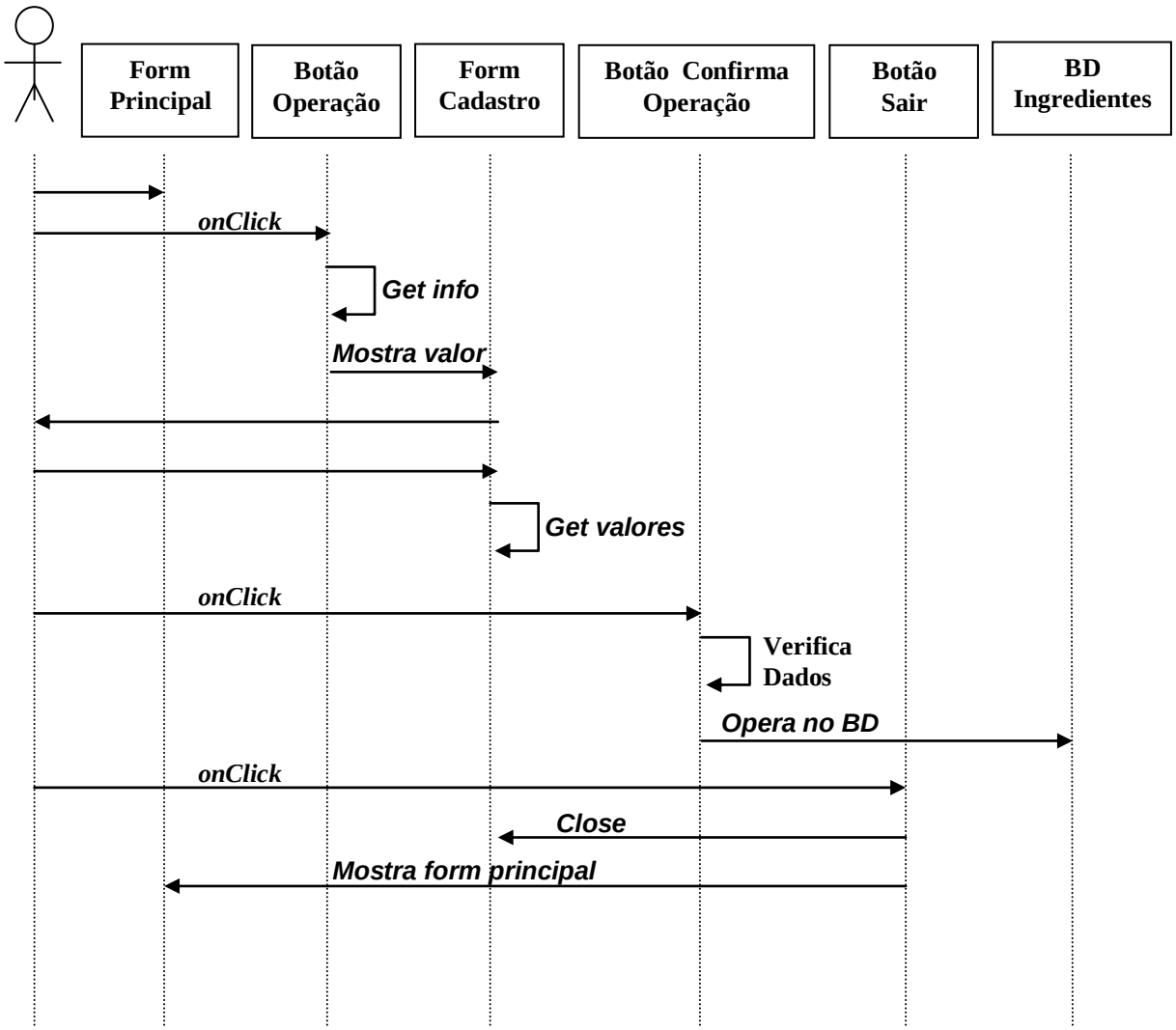


FIGURA 4.9 – Diagrama de Seqüência - cadastro de ingredientes

Na figura 4.10 é descrito todo o processo para que um funcionário seja cadastrado, alterado ou excluído do Banco de Dados. O gerente efetua uma chamada no botão de operação para fazer a chamada do cadastro e depois de escrito os valores ele verifica os dados e confirma a operação na tabela de funcionários.

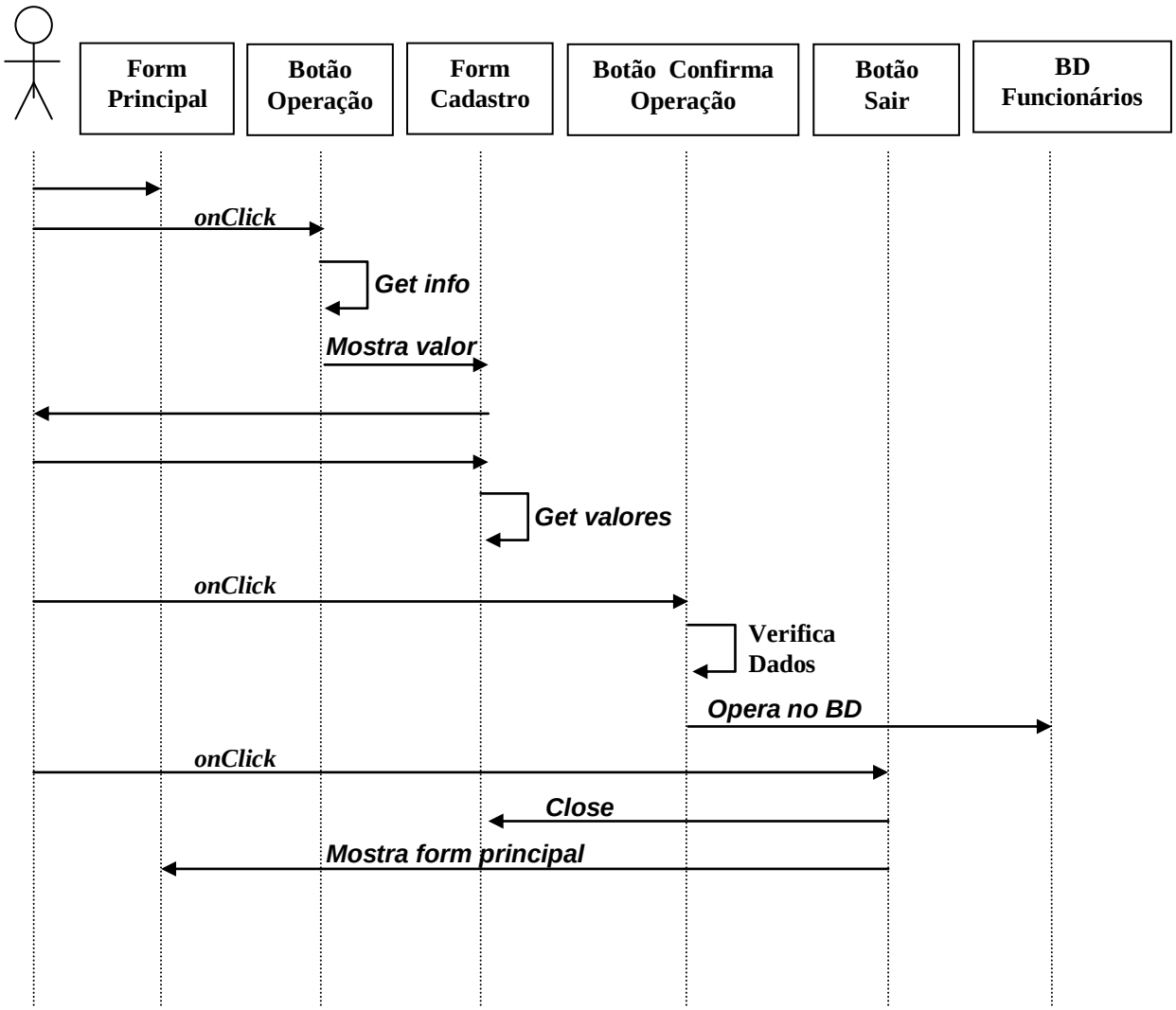


FIGURA 4.10 – Diagrama de Seqüência - cadastro de funcionários

Na figura 4.11 é descrito todo o processo para que os pedidos sejam cadastrados, alterados ou excluídos do Banco de Dados. O funcionário e/ou gerente efetua uma chamada no botão de controle de pedidos para fazer a chamada do formulário de controle e depois de escrito os valores ele verifica no Banco de Dados e confirma a operação e os dados ficam a disposição para consultas.

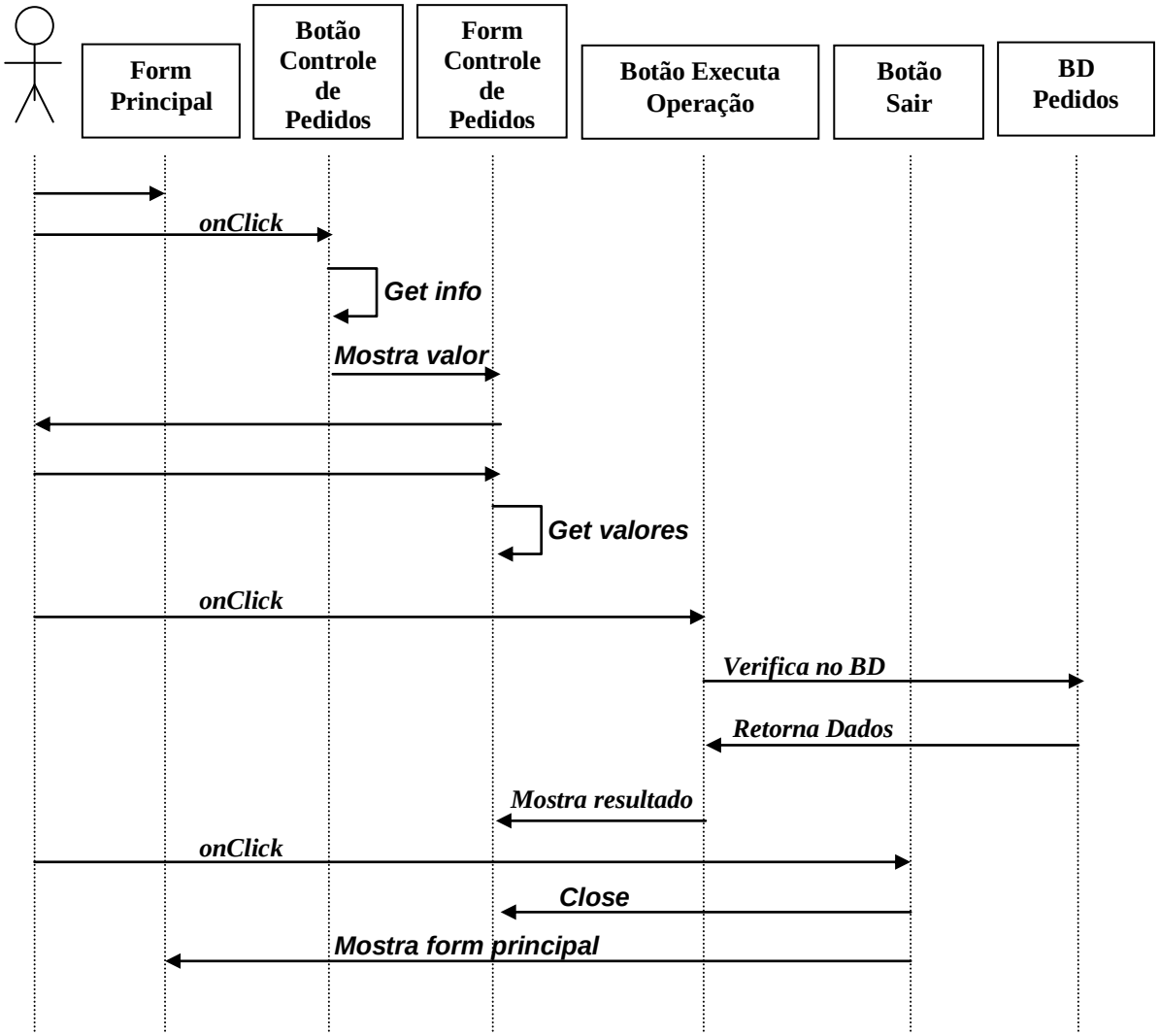


FIGURA 4.11 – Diagrama de Seqüência - controle de pedidos

Na figura 4.12 é mostrado o Diagrama de Entidade e Relacionamento. Este diagrama tem por base a percepção do mundo real no qual é formado por um conjunto de objetos, que são as entidades e pelo conjunto de relacionamentos entre esses objetos. Este diagrama foi desenvolvido para facilitar o projeto do Banco de Dados, permitindo a especificação do esquema de uma empresa, por exemplo, representando toda a parte lógica do Banco de Dados.

A partir das tabelas e atributos especificados, pode-se definir as ordem dos relacionamentos e as chaves primárias e estrangeiras de cada tabela, como mostra a figura abaixo.

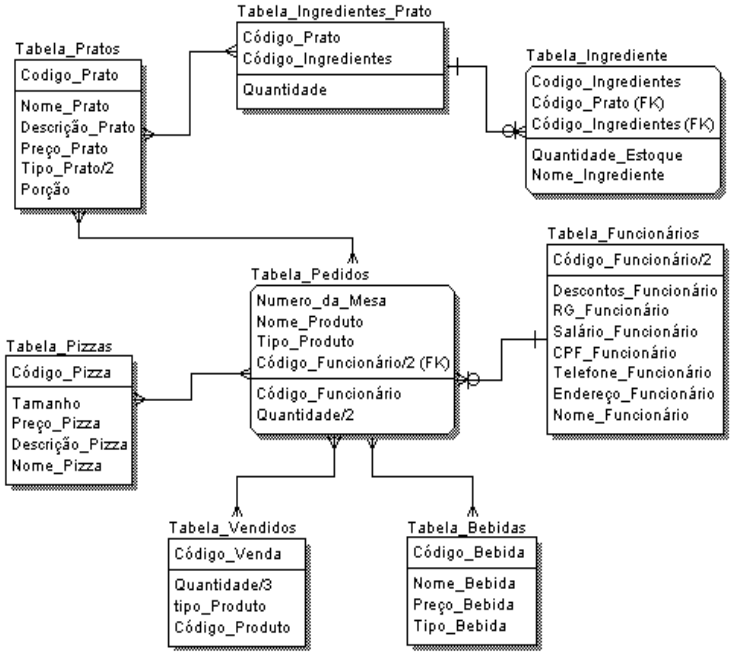


FIGURA 4.12 – Diagrama Entidade / Relacionamento

5. APRESENTAÇÃO DO SISTEMA

O sistema em questão foi desenvolvido no ambiente de programação Delphi, o qual se baseia na linguagem Pascal por objetos (Object Pascal). Este possui componentes que facilitam o processo de criação de um sistema desde o projeto de sua interface até a ligação com um Banco de Dados.

Para o armazenamento dos dados foi utilizado o Banco de Dados Interbase, um Banco de Dados robusto e compatível com a linguagem escolhida.

O sistema em questão realizará o controle de uma pizzaria (controle interno). Este sistema objetiva cadastrar todos os produtos existentes e, a partir disto, gerar consultas, relatórios e, principalmente, fazer o controle dos pedidos e geração de contas.

A figura 5.1 apresenta a tela principal do sistema. É a partir desta interface que serão inicializadas todas as operações de cadastro de pizzas, pratos, bebidas, ingredientes e funcionários.

No menu de movimentação são efetuados pedidos, geradas contas e verificado os produtos vendidos. As consultas podem ser realizadas por pedidos, produtos e funcionários. Por fim, no menu de relatórios são disponibilizados as informações dos produtos em estoque, vendidos entre outros.

A figura 5.2 apresenta uma tela para o cadastro de pizzas. Dentro desta interface, o usuário pode incluir, alterar, excluir pizzas e atualizar suas informações no Banco de Dados. Nesta tela ainda, o usuário pode verificar através do menu verifica, as pizzas já cadastras.

Além deste cadastro, o sistema possui ainda: cadastro de pratos, cadastro de ingredientes, cadastro de bebidas, cadastro de funcionário, e controle de ingredientes, com interfaces semelhantes à da figura 5.2.

FIGURA 5.1 – Tela Principal do Sistema

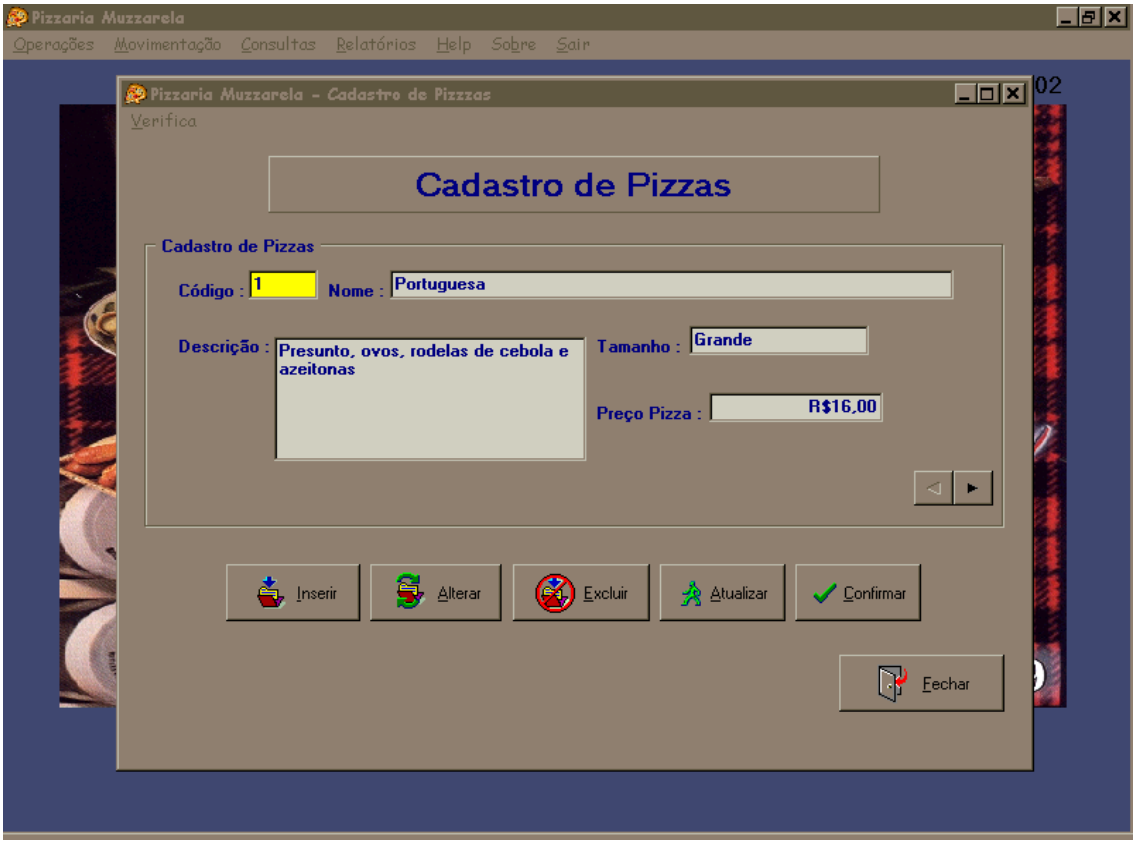


FIGURA 5.2 – Tela para Cadastro de Pizzas

A figura 5.3 mostra uma tela para efetuar o cadastro dos pedidos referentes a cada mesa. Nesta interface o usuário cadastra, altera, exclui e atualizar pedidos, além de poder verificar os pedidos cadastrados. Para que o pedido seja efetuado é necessário que o funcionário informe o número da mesa, o tipo do produto, nome e quantidade.

FIGURA 5.3 – Tela para Efetuar Pedidos

No menu de movimentação localizado na tela principal do sistema existe também a tela para a geração da conta.

A figura 5.4 mostra uma tela para consultar pedidos referentes a cada mesa. Nesta tela o usuário pesquisa por pedido, informando o numero da mesa para que os produtos consumidos sejam mostrados na tabela na parte inferior da tela. Nesta mesma tela o gerente ou funcionário poder gerar a conta, clicando no botão localizado na parte inferior da tela de consulta de pedidos.

No menu de consultas localizado na tela principal do sistema existem também as telas para a consulta de funcionários (código e nome) e produtos (ingredientes, pratos, pizzas e bebidas).

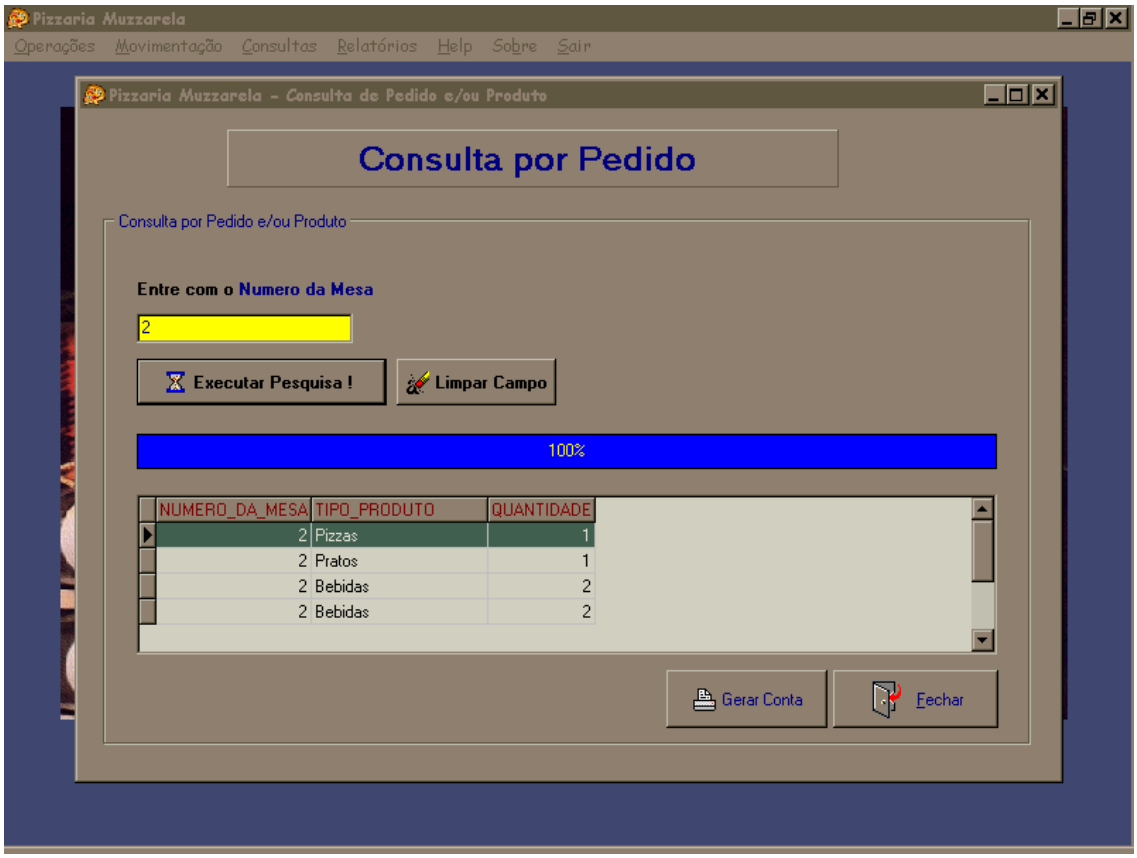


FIGURA 5.4 – Tela para Consultar Pedido

A figura 5.5 mostra uma tela para consultar produtos, através de seu código. As consultas são por ingredientes, pratos, pizzas e bebidas. Os resultados são mostrados em uma tabela, localizada na parte inferior da tela.

A figura 5.6 mostra uma tela de relatório de bebidas. Nesta tela o usuário observa todos os tipos de bebidas cadastradas no Banco de Dados, cada uma com seu código, nome, tipo e preço. O usuário pode salvar, fechar ou imprimir o relatório.

No menu de relatórios localizado na tela principal do sistema existem também outros relatórios como: relatório de pizzas, carnes, calzones, lasanhas, saladas, vinhos nacionais, vinhos importados entre outros.

Alem destes, serão disponibilizados relatórios de pizzas, pratos e bebidas mais vendidas. Esta tela será representada por gráficos de pizza para uma melhor visualização.

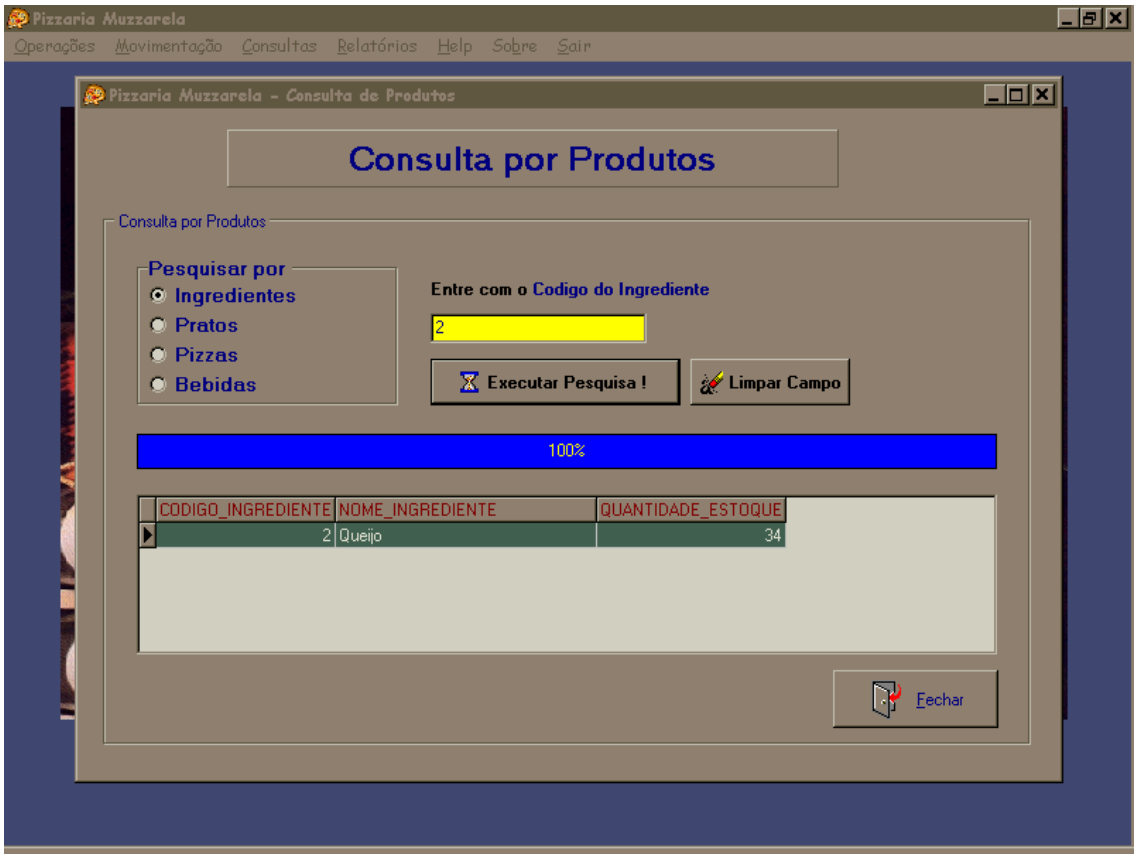


FIGURA 5.5 – Tela para Consultar Produtos

FIGURA 5.6 – Tela de Relatório de Bebidas

A Figura 5.7 mostra uma tela para gerar a conta final dos pedidos. Após digitado o numero da mesa, os resultados são mostrados em uma tabela na parte inferior da tela, contendo os campos: código do produto, nome do produto, preço do produto, quantidade e tipo do produto. Ao mesmo tempo que é gerado a conta, é feita a soma total dos pedidos, ou seja, o preço do produto vezes a quantidade pedida, e é mostrada num campo localizado abaixo da tabela.

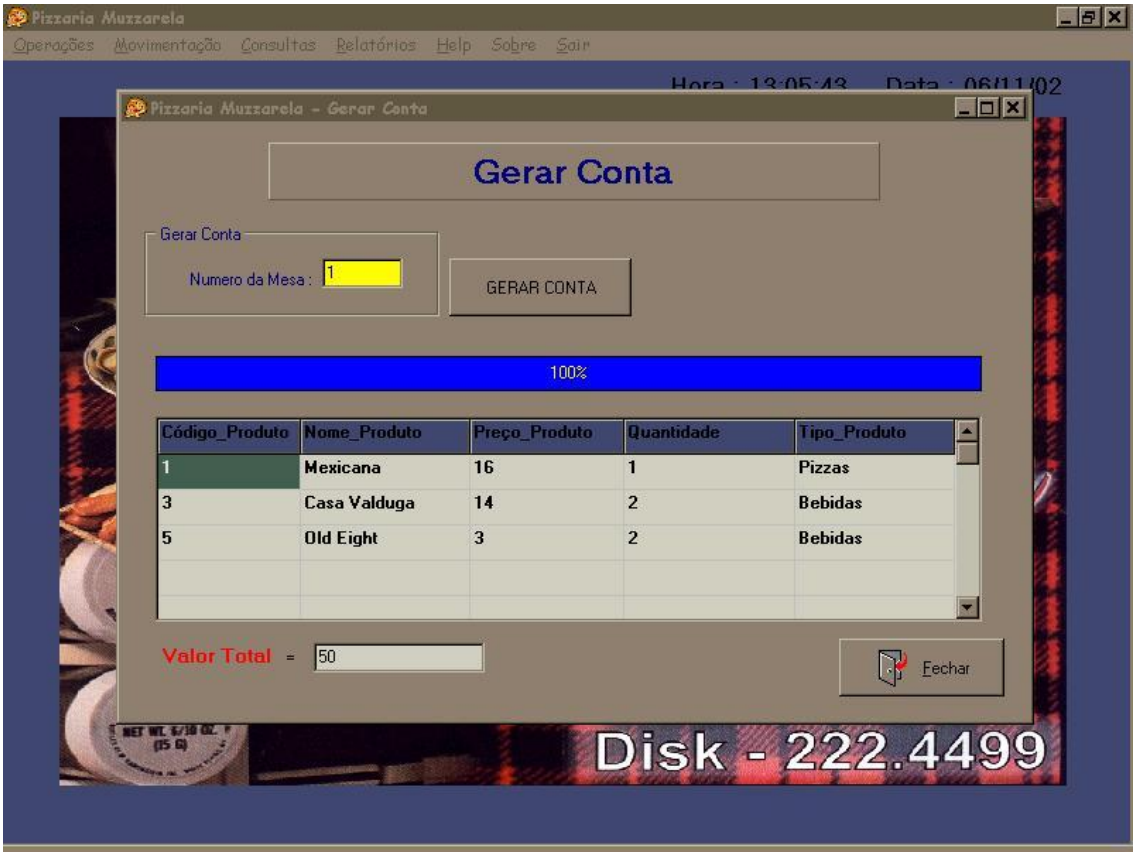


FIGURA 5.7 – Tela de Gerar Conta

A Figura 5.8 apresenta de forma gráfica as pizzas mais vendidas. Em cada fatia do gráfico, é representada uma pizza e sua percentagem de venda. Para melhor visualização foi criado uma legenda, localizada ao lado direito do gráfico, no qual é denotada por cores e nome das pizzas. Além do gráfico das pizzas, existem outros como pratos e bebidas mais vendidas localizado no menu de relatórios da tela principal do sistema.

Na Figura 5.9 apresenta uma tela de ajuda. É mostrado para cada menu do sistema, seus campos, operações, atalhos e como operar em cada uma delas.

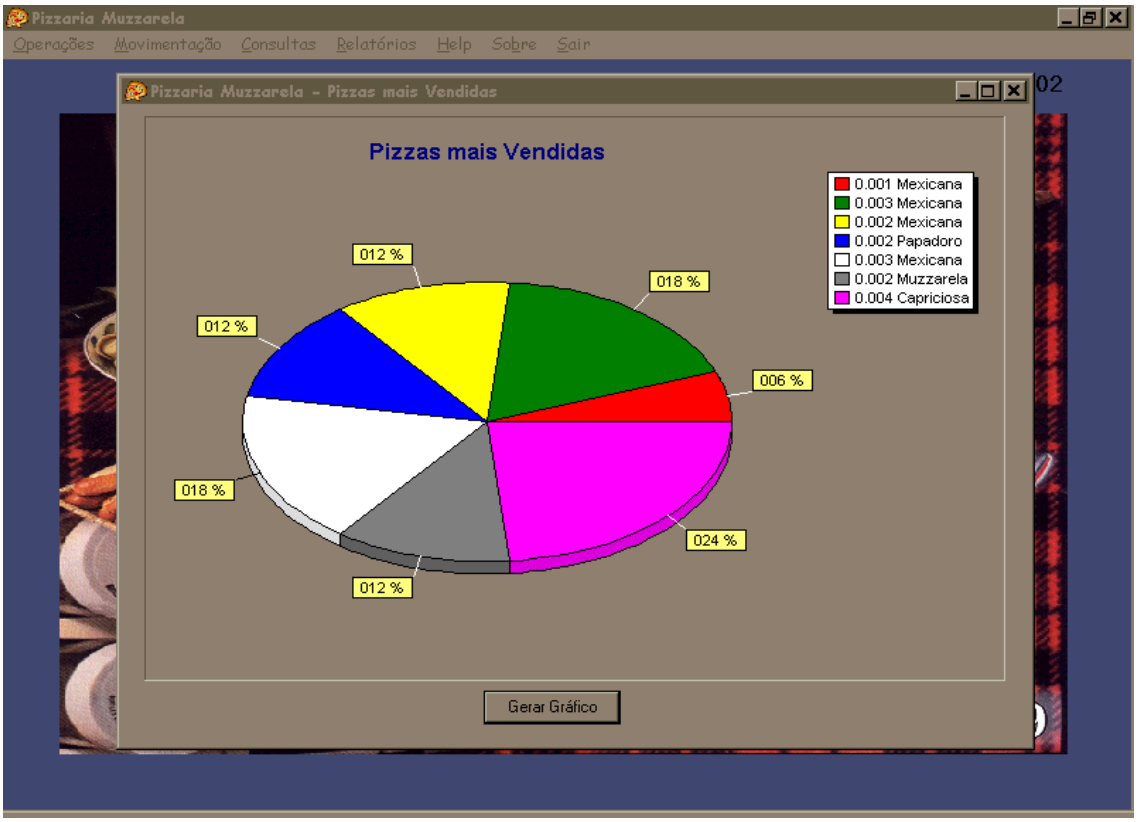


FIGURA 5.8 – Tela Pizzas mais Vendidas

FIGURA 5.9 – Tela de Ajuda

6. CONCLUSÃO

A divisão do processo de desenvolvimento em etapas facilita bastante a criação do sistema. Ela permite que o desenvolvedor inicie de uma descrição abstrata do problema e gradualmente migre para representações mais detalhadas do mesmo, até chegar ao código fonte. Por isso, todo este processo que antecede a geração do código final é de extrema importância para o sistema. Visando antes de tudo a qualidade principalmente no que se refere a forma de armazenamento dos dados.

Os diagramas estudados mostraram-se capazes de atender as necessidades do sistema em questão, pelo fato de ilustrar de forma clara os objetos e ligações decorrentes de cada operação do sistema. Outros diagramas não foram utilizados devido às características deste trabalho.

Os conceitos relacionados a desenvolvimento de sistemas e Banco de Dados, trouxeram uma melhor facilidade para a efetivação do trabalho tanto para o projeto como para a implementação do sistema.

A modelagem do sistema foi uma fase bastante importante, principalmente no momento da implementação. Com os requisitos levantados e especificados através dos modelos da UML facilitou-se à visualização das funções oferecidas pelo sistema.

Para a definição dos requisitos é importante destacar a série de entrevistas feitas com o cliente onde buscou-se o maior número de informações possíveis para que o sistema resultante atingisse os objetivos desejados.

Além disto, a análise do sistema atual ajudou a esclarecer pontos que não estavam bem claros após estas entrevistas.

A versão atual do sistema, já facilita o controle dos pedidos e geração de contas referentes a cada mesa. Principalmente por sua interface amigável, tornando

então de fácil manuseio. Com base no que já existe e o que pode ser feito no sistema concluí-se que vai melhorar significativamente todo o controle da pizzaria.

Para dar continuidade a este trabalho propõe-se o desenvolvimento de um módulo automático para o controle de estoque, cuja função seria avisar ao gerente quando algum produto estiver abaixo de uma cota mínima. Além disso técnicas de mineração de dados podem ser estudadas e aplicadas para prever a tendência das vendas e ajudar o gerente a planejar melhor a compra de ingredientes.

Outra proposta para trabalhos futuros, seria adaptar o sistema para rodar via internet, ou seja, pizzaria on-line.

7. REFERÊNCIAS

BLUE, T.; KASTER J.; LIEF G.; SCOTT L. **Desenvolvendo Banco de Dados em Delphi 3.0**. São Paulo: Makron Books, 1998. 776 p.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML : Guia do Usuário**. Rio de Janeiro: Campus, 2000. 472 p.

CARVALHO, A. M. B. R.; CHIOSSI, T. C. dos S. **Introdução a Engenharia de Software**. Campinas: Unicamp, 2001. 148 p.

MACHADO, F. N. R.; ABREU, M. **Projeto de Banco de Dados : Uma Visão Pratica**. São Paulo: Érica, 1996. 298 p.

MATOS, V. de M. **UML : Prático e Descomplicado**. São Paulo: Érica, 2002. 187 p.

PRESSMAN, R. S. **Software Engineering : A Practitioner´s Approch**. New York: McGraw-Hill, 2001. 160 p.

PRESSMAN, R. S. **Engenharia de Software**. São Paulo: Makron Books, 1995. 1056 p.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. **Sistema de Banco de Dados**. 3. ed. São Paulo: Makron Books, 1999. 778 p.

SILVA, N. P. **Projeto e Desenvolvimento de Sistemas**. São Paulo: Érica, 1994. 143 p.

TEIXEIRA, S.; PACHECO, X. **Delphi 5 : Guia do Desenvolvedor**. Rio de Janeiro: Campus, 2000. 1219 p.