Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

MARCELO GONSO DE LIMA

TECNOLOGIAS LINQ PARA DESENVOLVIMENTO DE APLICATIVO UTILIZANDO LINGUAGEM C#

ASSIS
2010

Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

MARCELO GONSO DE LIMA

TECNOLOGIAS LINQ PARA DESENVOLVIMENTO DE APLICATIVO UTILIZANDO LINGUAGEM C#

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação.

Orientador: Prof.Dr. ALEX SANDRO ROMEO DE SOUZA POLETTO

Área de Concentração: Linguagem de Programação.

Assis
2010

FICHA CATALOGRÁFICA

LIMA, Marcelo Gonso
Tecnologias Linq para desenvolvimento de aplicativo utilizando linguagem c#/ Marcelo Gonso de Lima. Fundação Educacional do Município de Assis – FEMA – Assis, 2010.
56p.

Orientadora: Prof. Dr. Alex Sandro Romeo de Souza Poletto
Trabalho de Conclusão de Curso – Instituto Municipal de Ensino Superior de Assis – IMESA.

1.C#. 2.Linq. 3.Banco De Dados.

CDD:001.61
Biblioteca da FEMA.

Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

TECNOLOGIAS LINQ PARA DESENVOLVIMENTO DE APLICATIVO UTILIZANDO LINGUAGEM C#

MARCELO GONSO DE LIMA

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito de Curso de Tecnologia em Processamento de Dados, analisado pela seguinte comissão examinadora:

Orientadora: Prof.Dr. ALEX SANDRO ROMEO DE SOUZA POLETTO

Analisador : Prof. FERNANDO CESAR DE LIMA

ASSIS
2010

DEDICATÓRIA

Dedico este trabalho, à minha querida mãe Maria Helena Gonso de Lima (in memoriam) que por tanto fazer o seu melhor até hoje vivemos de seus frutos, a meu pai Célio Carvalho de Lima que mesmo contido fez transparecer seu amor por seus filhos, a meus avôs maternos por me guardarem e vigiarem, a meu tio Silvio Carvalho de Lima (in memoriam) que foi meu maior incentivador para o ingresso na faculdade.

AGRADECIMENTOS

Agradeço primeiramente a Deus, que sempre me atendeu em meus pedidos, dando força e paciência para concluir mais esta etapa de minha vida;

À minha mãe que mesmo não estando mais presente entre nós, deixou e garantiu todo o suporte financeiro para que pudesse concluir este curso;

Ás mulheres que sempre estiveram presente em todos os momentos de minha vida que são minhas avós, mães e avós de amigos e tias que me adotaram como filho na ausência de minha mãe, e tenho certeza que não seria o mesmo sem vocês;

Ao meu Pai Célio Carvalho de Lima, ao meu avô José Gonso, a minha avó Ernestina do Carmo Gonso, ao meu irmão Cristiano Gonso de Lima e a minha namorada Marcela de Lima Claudio por existirem e por estarem perto em todos os momentos;

Ao meu orientador, Prof. Dr. Alex Sandro Romeo de Souza Poletto, a quem devoto a mais sincera admiração, por toda sabedoria, apoio e incentivo para a conclusão deste trabalho e também por sempre compartilhar seu conhecimento e seu tempo durante todo o curso;

Aos meus professores Begosso, Campanati, Célio, Dani, Douglas, Domingos, Fernando, Guto, Laudo, Leonor, Osmar e Sara por passarem todos os conhecimentos de informática, principalmente a Marisa por nunca deixar que seus alunos desistam e por nunca nos esquecer, e o seu anonimato é recompensado em resultados, ao Almir por ser o pai que eu gostaria de ter e por me ensinar do jeito mais didático, isso mostra que o profissional pode ter coração, ao Alex por me fazer sonhar com o futuro e me ensinar o ramo de TI que escolhi para ser profissional e ao Talo por ser a base de tudo que aprendi, mostrando que o inicio é pequeno, mas o fim é grandioso;

À Fundação Educacional do Município de Assis por ter-me possibilitado a realização deste trabalho;

Aos colegas de curso e aos grandes amigos pelo carinho, compreensão e força que sempre me deram e por sempre estar juntos nos momentos mais importantes e poder “contar” com vocês sempre será meu refugio;

E a todos aqueles que, direta ou indiretamente, colaboraram para a realização deste trabalho e ter conseguido atingir os objetivos propostos.

Durante este trabalho...

Desisti recomecei lutei consegui
Não tenhais medo do que ainda não viveu
Não se doa por algo que ainda não dói
Não desista antes do fim

Pois se há algum objetivo
Não há obstáculo que não seja intransponível
Sei que a partir de hoje começa meu futuro

Não tenhais medo do que ainda não viveu
Sei que farei falta onde estou
Mas vou, pois algo me chama
Por que sei que esse algo se chama sucesso.

Marcelo Gonso de Lima

RESUMO

Com um mercado pujante o mundo dos Bancos de Dados é cada vez mais atrativo e necessário, pois, quem detém o maior número de dados é quem mais ganha com isso. Porem, dados são apenas dados, quando não empregados, interagi-los com um programa é ainda mais valoroso, pois ao dar agilidade ao processo de armazenagem de informações, pode-se ditar o sucesso de uma ferramenta ou o desempenho de um programa. Para tanto, este trabalho apresenta a especificação e a implementação de um aplicativo desenvolvido em C# do pacote .NET FRAMEWORK. Com isso fazer a interação Banco X Aplicativo mediante tecnologia LINQ, com o intuito de agilizar administração de dados cadastrais de uma loja de perfumes. Será feita uma abordagem da descrição teórica sobre LINQ, uma vez que essa ferramenta é amparada pelo peso da marca Microsoft, e muito se espera em relação à produtividade. Neste trabalho será desenvolvido um aplicativo utilizando estas tecnologias para o setor de perfumarias.

Palavras Chave: LINQ; C#; Banco de Dados.

ABSTRACT

With a booming market the world of databases is becoming increasingly attractive and necessary, because who has the largest number of data is the big winner from this. But data is only used when no data, interacted with them a program is even more valuable as giving flexibility to the process of storing information, one can dictate the success of a tool or a program's performance. Therefore, this paper presents the specification and implementation of an application developed in C # package. NET FRAMEWORK. With this Database X Application to the interaction through LINQ technology, in order to streamline administration of register data in a Fragrance store. There will be a theoretical approach to the description on LINQ, since this tool is supported by the weight of the Microsoft brand, and much is expected in relation to productivity. In this work we developed an application using these technologies to the industry of perfumery.

Keywords: LINQ, C #, Database.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

SGBD	Sistema Gerenciador de Banco de Dados
SBD	Sistema de Banco de Dados
DDL	Data Definition Language
DML	Data Manipulation Language
DBA	Administrador de Banco de Dados
SQL	Structure Query Language
LINQ	Language Integrated Query
RDBMS	Relational Database Management System
XML	Extensible Markup Language
OMR	Object-Relational Mappin
EDM	Entity Data Model
DOM	Document Object Model

SUMÁRIO

1. INTRODUÇÃO

Durante muito tempo, existia um cenário real onde um desenvolvedor conhecia bastante sobre Banco de Dados e por outro lado um desenvolvedor que era especialista em orientação objeto (OO) onde de um lado ficava todo um conhecimento de orientação a objeto e de outro todo conhecimento de acesso a Dados entre toda essa informação se criava uma barreira de comunicação entre os dois cenários. Com isso tinha se códigos muito extensos e com muitas brechas para bugs, quebras de serviços, com as necessidades vieram às tentativas e a disputa de mercado já que a dificuldade virou um mercado. Diante da grande variedade de tecnologias e a enorme necessidade de soluções. Nos últimos anos as duas plataformas no caso Java e C# se estabeleceram como as principais alternativas para o mundo enterprise, correram atrás de inovações para causar diferenciais, causaram inúmeras discussões e até mesmo tiveram conversas agradáveis (DALTON, 2010)**.**

Fruto dessa disputa, mas também para fazer frente a um seguimento que toma grande parte do mercado de SGBD, com fama de programação ágil e desenvolvimento simples a Microsoft desenvolveu a *Language-Integrated Query* (LINQ) (CENTER, 2010) é o nome de um conjunto de tecnologias baseadas na integração de recursos de consulta direcionada para a Linguagem C# também para Visual Basic e potencialmente para qualquer outra linguagem .NET *framework* (MSDN, 2010). O LINQ é uma tecnologia de fácil aprendizagem e grande eficácia nas quais consultas, inserções, atualizações e consultas em coleções de objetos onde possam ser efetuadas sem que perca tempo com muitas linhas de código, portanto dar mais agilidade na burocracia em acesso a um banco de dados feito por um programa. E assim fazer com que as manipulações de Bancos de Dados trabalhem um pouco mais leves.

Este trabalho visa fundamentalmente adquirir conhecimentos das tecnologias LINQ utilizadas na *área de Programação*, com o intuito de desenvolver aplicativos para o entendimento da utilização da ferramenta.

1.1 Objetivos

O principal objetivo deste trabalho é adquirir conhecimento teórico do pacote .NET com ênfase na tecnologia LINQ e desenvolver um aplicativo C# com o uso desta ferramenta. Uma das principais funcionalidades do aplicativo é gerenciar os dados cadastrais de clientes e produtos, e elaborar o orçamento de produtos para uma loja de perfumes. Uma das vantagens de se utilizar LINQ é a capacidade de fornecer ao desenvolvedor os subsídios na consulta de dados.

1.2 Motivação

Uma das principais motivações para o desenvolvimento deste trabalho, sem dúvida alguma, é o crescente mercado de serviços LINQ. Esta tecnologia não é recente, porém muito produtiva, e nos últimos cinco anos tem sido muito explorada no meio profissional. Um dos problemas desta tecnologia é a falta de documentação mais detalhada, o que dificulta a sua utilização nos meios acadêmicos. O desenvolvimento do aplicativo com esta tecnologia será de grande importância, pois futuramente será utilizado para novos aplicativos.

1.3 Justificativa

A justificativa para o desenvolvimento deste trabalho deve se ao interesse em adquirir conhecimento de novas tecnologias relacionadas a Banco de Dados, devido à exigência do mercado de atuação. Isso pode ser observado pelos dados obtidos (FACTS, 2008), onde o mercado mundial de RDBMS (*Relational Database Management Systems*) se encontra bem pulverizado, por exemplo: Oracle (44,3%), IBM (21%) e Microsoft (15,8%).

O tema abordado é de alta relevância, tendo em vista que o LINQ dá suporte para todos os Banco de Dados desde que possuam *driver* para o ADO.NET *Entity*

Framework (HADDAD, 2009). A utilização da linguagem C# é devido a ser uma das poucas de interage com a tecnologia LINQ.

1.4 Estrutura do trabalho

Este trabalho está subdividido capítulos que serão explicitados a seguir.

O primeiro capítulo apresenta a contextualização e justificativa para o desenvolvimento da proposta do trabalho.

O segundo capítulo aborda a fundamentação teórica dos conceitos básicos que foram utilizados para o desenvolvimento e modelagem do projeto.

O terceiro capítulo apresenta o desenvolvimento do trabalho, mostrando a modelagem do problema, os diagramas de classe, casos de uso e diagramas de seqüência. Será feita uma descrição da implementação do aplicativo

O quarto capítulo apresenta a conclusão do trabalho.

2. FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, será feita uma fundamentação teórica dos conceitos básicos que foram utilizados para o desenvolvimento e modelagem do projeto. Estes conceitos se referem às linguagens e as ferramentas necessárias para o desenvolvimento do aplicativo.

2.1 Framework

É uma estrutura de suporte definida em que um projeto de software pode ser organizado e desenvolvido. Um framework pode incluir programas suporta bibliotecas de código, linguagens de script e outros softwares para auxiliar no desenvolvimento e unir diferentes componentes de um projeto de software. Existem, literalmente, milhares de frameworks disponíveis para as diversas linguagens de programação existentes. É importante que diante esta grande imensidão de frameworks ofertados no mercado de programação, optar por um que atenda a suas necessidades e que seja mais acessível seu método e estilo de trabalho.

Inicialmente, pode parecer que usar um *framework* é pior do que fazer uma “programação pura”. Entretanto, tenha certeza: vale à pena, pois. As vantagens vindas de um projeto bem estruturado e rodando numa plataforma segura e estável, realmente compensam. (ZEMEL, 2009)

2.1.1 .Net Framework 3.5

O .NET Framework 3.5 apresenta novos recursos para as tecnologias no 2.0 e no 3.0 e tecnologias adicionais na forma de novos *assemblies*. As seguintes tecnologias são introduzidas com o .NET *Framework* 3.5: Idioma Integrated Consulta (LINQ). Novos compiladores para C#, Visual Basic e C++.ASP.NET AJAX.A figura 1 mostra .Net 3.5 *Framework*.

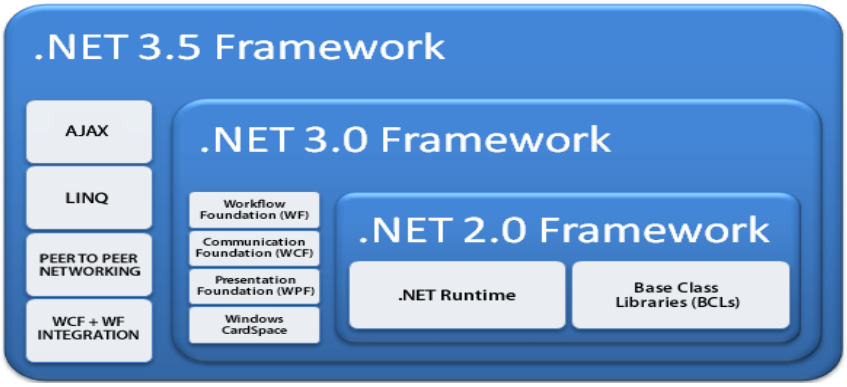

Figura 1: .Net 3.5 Framework

2.1.2 .Net Framework 4.0

O que não poderia deixar de citar, que Durante a elaboração deste trabalho a Microsoft lançou a plataforma. NET 4.0, como *MEF - Managed Extensibility Framework, DLR – Dynamic Language Runtime, EF – Entity Framework 4.0, PLINQ – Parallel LINQ*, além das novas versões *WCF4, WF4 e WPF4*. O *ASP .NET* e o *AJAX* também ganharam diversos recursos e melhorias na nova versão do framework., a figura 2 mostra a arquitetura .NET FRAMEWORK 4.0

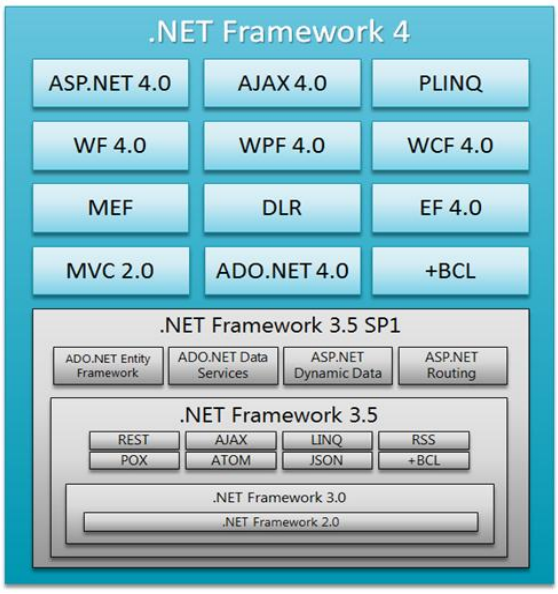

Figura 2: Arquitetura .NET FRAMWORK 4.0

2.1.2.1 O que é CLR

O CLR é um ambiente de tempo de execução (*runtime*) que realiza tarefas, tais como: execução do programa, gerenciamento de memória (coleta de lixo), segurança, tratamento de erro, controle de versão e suporte de instalação. Realiza a interface entre a aplicação e o sistema operacional. O código que é executado nesse ambiente de *runtime* é chamado de Código Gerenciado (“Managed Code”), enquanto aquele que é executado fora é chamado de Código Não Gerenciado (“Unmanaged Code”). A figura 3 mostra O CLR.

Figura 3: O CLR

2.1.2.2 O que é Base Class Library

A *Base Class Library* (BCL) é um conjunto de classes que o .NET disponibiliza para todas as linguagens que rodam sob o .NET Framework. Essa base encapsula várias funcionalidades que tornam o trabalho do desenvolvedor muito mais fácil e qualquer linguagem do *Framework* pode utilizar.

A BCL é organizada em *namespaces*, que são grupos de classes relacionadas (SALES, 2010). A figura 4 mostra Entendendo BCL.

Figura 4: Entendendo BCL

2.2 Linguagem C#

C# (Csharp) é uma linguagem de programação orientada a objetos criada pela Microsoft, faz parte da sua plataforma .Net. A companhia baseou C# na linguagem C++ e Java. Originalmente desenvolvida por um pequeno time conduzido por dois engenheiros da Microsoft, Anders Hejlsberg e Scott Wiltamuth. Hejlsberg é também conhecido pela criação do Turbo Pascal, uma linguagem popular para Programação PC, e por liderar o time que arquitetou o Borland Delphi, um dos primeiros vitoriosos Ambientes de Desenvolvimento Integrados (*Integrated Development Environments*, IDEs) para programação cliente/servidor. Foi criada praticamente do zero para funcionar na nova plataforma, sem preocupações de compatibilidade com código de legado. O compilador C# foi o primeiro a ser desenvolvido. A maior parte das classes do .NET Framework foram desenvolvidas em C# (EDUCACAO, 2008).

2.2.1 O que é C#

A sintaxe do C# é altamente expressiva, mas ela também é simples e fácil de aprender. C# será instantaneamente reconhecida por qualquer pessoa familiarizada com C, C++ ou Java. Os desenvolvedores que sabem qualquer uma dessas linguagens são geralmente capazes de começar a trabalhar de forma produtiva com C# dentro de um tempo muito curto. Sua sintaxe do C# simplifica muita das complexidades do C++ e fornecem recursos poderosos, como tipos de valor nulo, enumerações, delegados, expressões lambda e acesso direto a memória, que não são encontrados no Java. O C# suporta métodos e tipos genéricos, que fornecem uma melhor segurança de tipo e desempenho, e iteradores, que permitem implementadores de coleções de classes para definir comportamentos de iteração personalizados que são simples de usar pelo código cliente. LINQ (consulta integrada à linguagem) expressões de fazer com que a consulta fortemente tipado uma construção de linguagem de Primeiro classe.

Como é uma linguagem orientada a objetos, o C# suporta os conceitos de encapsulamento, herança e polimorfismo. Todas as variáveis e métodos, incluindo o método principal (*Main*), o ponto de execução de uma aplicação, é encapsulado em definições de classes. Uma classe derivada pode herdar diretamente somente de uma classe pai, mas pode herdar de qualquer quantidade de interfaces. Métodos da classe derivada que substituem métodos virtuais de uma classe pai exigem a utilização da palavra-chave *override* como forma de evitar a redefinição acidental. Em C#, uma *struct* é como uma classe simplificada; é um tipo alocado em pilha que pode implementar interfaces mas não suporta herança (MSDN, 2010). A figura 5 mostra C# em Ação.

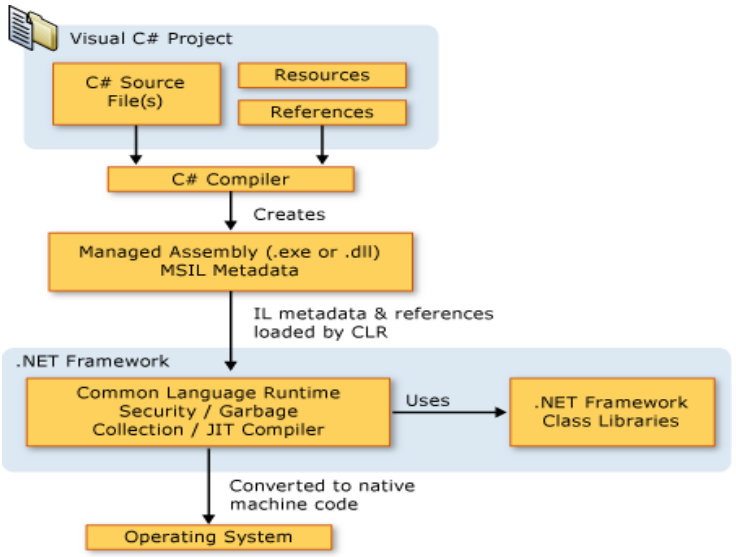

Figura 5: C# em Ação

2.3 Histórico Do LINQ

Matt Warren postou em seu blog, um pouco da historia do LINQ informalmente, tudo começou no outono de 2003 muito antes de alguém ouvir falar sobre LINQ ele era apenas um humilde projeto do Visual Studio na sua máquina desktop onde ele detinha o C# *codebase* e tinha um desafio de transformá-lo em uma linguagem que poderia realmente fazer algo interessante para uma mudança, uma linguagem que soubesse como consultar.

Para isso ele precisava de árvores de expressão e um SQL tradutor (WARREN, 2007). Porem o LINQ até então seria parte de outro produto da Microsoft o WinFs, que viria com uma outra ORM ,mas com o fim do projeto ou como Matt Warren prefere dizer com o fiasco do WinFs .

LINQ um produto que havia tido ótimo desempenho nos seus testes precisava de mais que resultado para ter continuidade e como seu ORM fora moldado em conjunto com o C# 3.0 o LINQ que outrora fora um humilde projeto de um produto Microsoft, teve a partir de sua inserção no projeto C# 3.0, então depois de aceito no projeto, começa a defesa política de seus desenvolvedores dentro da Microsoft, pois

uma coisa é inserir um projeto dentro de um produto piloto e outro é fazer com que seus diretores o aceitem como produto comercial.

No entanto o projeto LINQ foi aceito e incorporado .NET Framework. Para permitir o suporte foi incorporado ao .NET algumas novas funcionalidades de forma a suportar o LINQ (Linguagem Integrada de Consulta). Dentre as novas funcionalidades, serão citadas: Tipos anônimos, expressões lambdas e extensões de métodos, além da *query* de consulta (DURÃES, 2007).

Que significa total adesão ao projeto LINQ uma vez que ele não precisa ser instalado separadamente ao instalar o Visual Studio 2008 ou os próximos que virem aparecer no mercado, terá em seu pacote além da linguagem C# e Visual Basic também todo o conjunto que engloba a LINQ que são: LINQ TO SQL, LINQ TO ENTITIES, LINQ TO DATASET, LINQ TO OBJECTS, LINQ TO XML.

2.3.1 Conceitos do LINQ

A LINQ define um conjunto de operadores padrões de consulta geral que permite que operações de travessia, de filtragem e de projeção sejam expressas do modo direto, porém declarativo, em qualquer linguagem de programação baseada em .NET. As criações dos processos de consulta sobre dados relacionais (DLINQ) são baseados na integração de definições de esquema SQL no sistema de tipos da CLR. Essa integração fornece tipagem forte sobre os dados relacionais, mantendo ainda o expressivo poder do modelo relacional e o desempenho na avaliação de consultas feitas diretamente no sistema de armazenamento. A extensibilidade da arquitetura de consulta é utilizada no próprio projeto LINQ para fornecer implementações que funcionam tanto sobre dados XML quanto SQL (BOX, HEJLSBERG , 2005). A figura 6 mostra a arquitetura do LINQ.

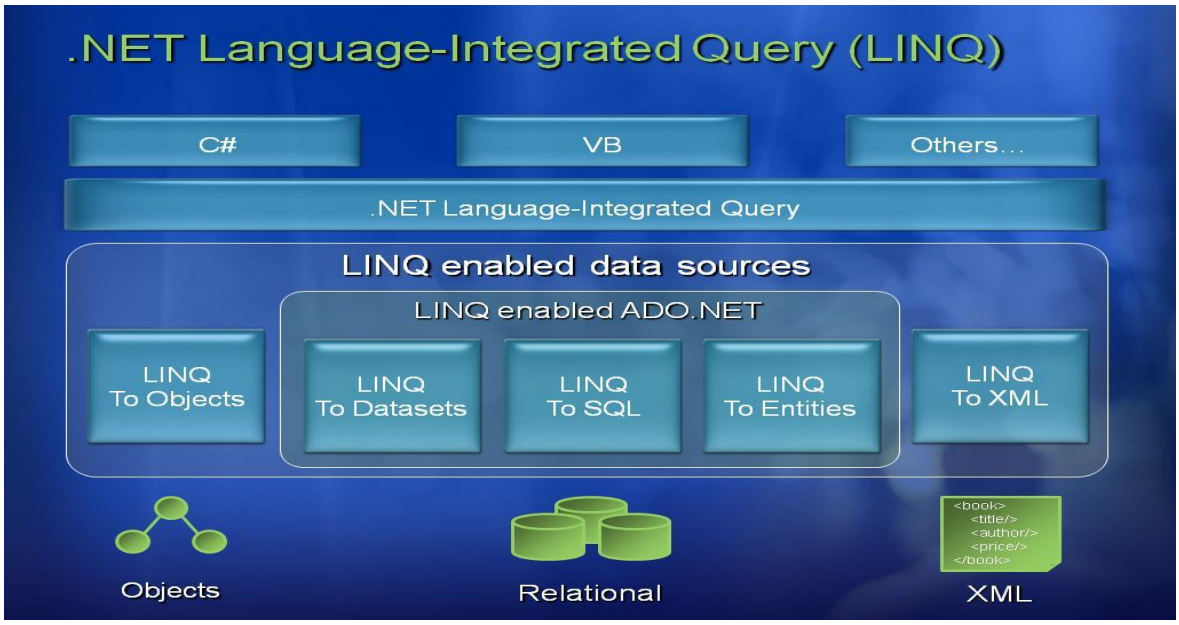

Figura 6: Arquitetura do LINQ

2.3.2 Características do LINQ

LINQ tem como característica sua grande abrangência, pois é capaz de fornecer ao desenvolvedor amparo a tudo que se relaciona a consulta de dados, como mostrado na arquitetura do LINQ da figura 6. Será feita uma descrição de cada extensão do LINQ.

2.3.2.1 Sistema De Banco De Dados

Os sistemas de banco de dados são projetados para gerenciar grandes blocos de informações. Esses grandes blocos de informações não existem isolados. Eles são parte da operação de alguma empresa cujo produto final pode ser informações do banco de dados ou pode ser algum dispositivo ou serviço para o qual o banco de dados desempenha apenas um papel de apoio (SILBERSCHATZ, 2006).

Um Banco de Dados é uma coleção de dados armazenados e inter-relacionados, que atende às necessidades de vários usuários dentro de uma ou mais

organizações, ou seja, coleções inter-relacionadas de muitos tipos diferentes de tabelas (TEOREY, 2007).

2.3.2.2 LINQ to SQL

É um mapeamento objeto-relacional (ORM), estrutura que permite o mapeamento 1-1 direto de um banco de dados Microsoft SQL Server para classes .NET, consulta e dos objetos resultante usando LINQ. Mais especificamente, o LINQ to SQL foi desenvolvido para atingir um cenário de desenvolvimento rápido contra Microsoft SQL Server onde o banco de dados se assemelha ao modelo de objeto do pedido e a principal preocupação é o aumento da produtividade do desenvolvedor (MSDN, 2010). A figura 7 mostra LINQ To SQL.

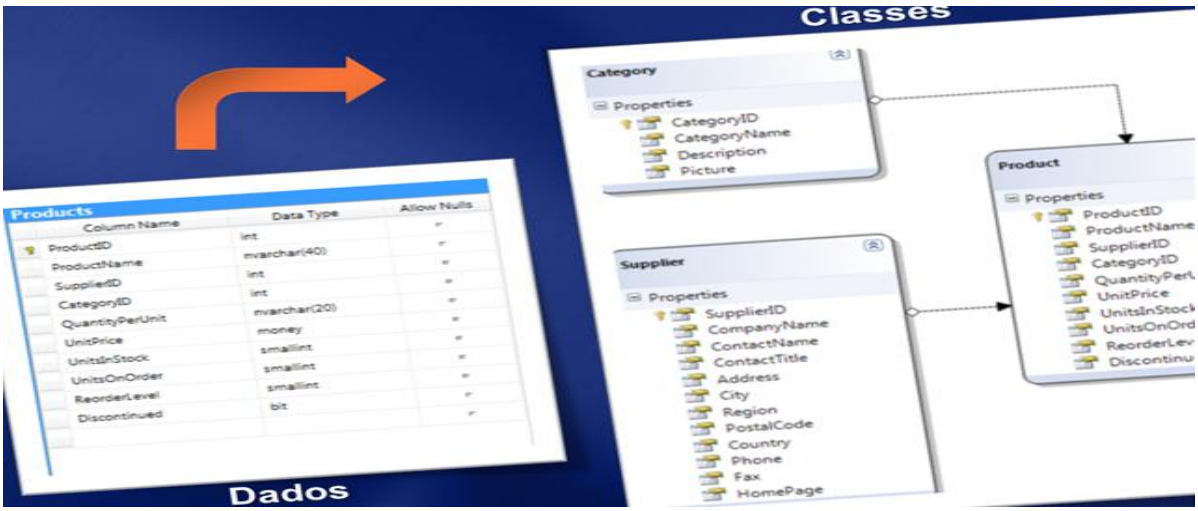

Figura 7: LINQ To SQL

2.3.2.3 Entity Data Model

O *Entity Data Model*- EDM é um modelo conceitual de dados que pode ser usado para modelar os dados de um domínio particular de forma que as aplicações podem interagir com os dados como se fossem entidades ou objetos. O conceito central no EDM são as entidades que são instâncias de tipos de entidades, por exemplo:

Cliente, Produto, etc. As quais são registros estruturados com uma chave. Uma entidade chave é formada a partir de um subconjunto de propriedades de uma entidade tipo; por exemplo: *Cliente ID, Produto ID*, etc. e é um conceito fundamental para identificar de forma única e assim atualizar instâncias de entidades e permitir que elas participem dos relacionamentos.

Aplicações que usam o EDM definem entidades e relacionamento no domínio da aplicação em um esquema que é usado para construir classes programáveis que são usadas pelo código da aplicação (MACORATTI, 2010).

2.3.2.4 LINQ to Entities

É, precisamente, uma parte do ADO.NET *Entity Framework* que permite que recursos de consulta LINQ. O *Entity Framework* é a evolução do ADO.NET que permite aos desenvolvedores programar em termos de captação ADO.NET padrão ou em termos de objetos persistentes (ORM) e é construído sobre o modelo padrão do ADO.NET provedor que permite o acesso de terceiros as bases de dados. O *Entity Framework* introduz um novo conjunto de serviços ao redor do *Entity Data Model* (EDM) (a média para a definição de modelos de domínio para uma aplicação) (MSDN, 2010). Através do EDM a ADO .NET expõe as entidades como objetos no ambiente .NET o que torna a camada de objetos um destino ideal para o suporte a LINQ. É por isso que LINQ to ADO .NET inclui LINQ to Entities.

O recurso LINQ To Entities permite que o desenvolvedor escreva consultas contra um banco de dados a partir da mesma linguagem usada para construir a lógica de negócio (MACORATTI, 2010). A figura 8 mostra LINQ To Entities.

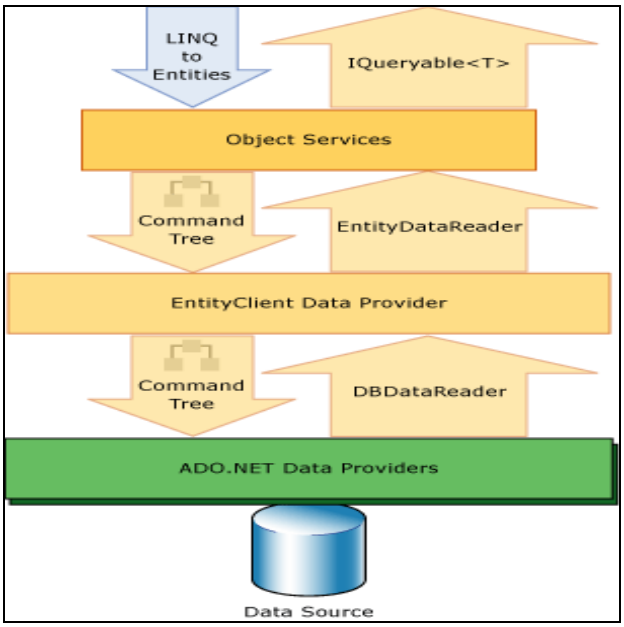

Figura 8: LINQ To Entities

2.3.2.5 Dataset ADO.NET

O *Dataset* ADO.NET é expressamente concebidos para o acesso aos dados independente de qualquer fonte de dados. Como resultado, ele pode ser usado com múltiplas e diferentes fontes de dados, utilizado com dados XML, ou usadas para gerenciar dados locais para a aplicação. O *DataSet* contém uma coleção de um ou mais *DataTable* objetos composta por linhas e colunas de dados, bem como chave primária, chave estrangeira, e a relação de informações sobre os dados no *DataTable* objetos.O diagrama abaixo ilustra a relação entre a *Framework*. Provedor de dados e um *Dataset.* A figura 9 mostra Arquitetura ADO.NET.

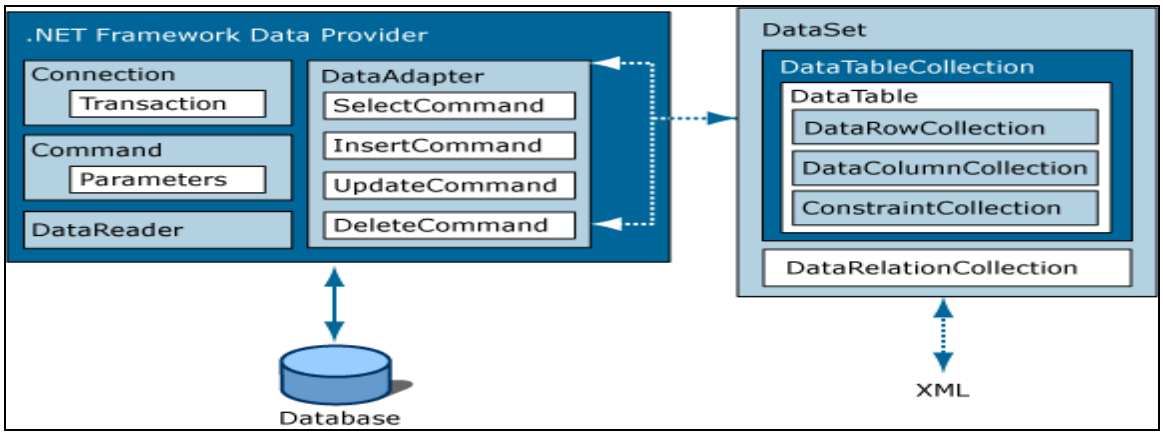

Figura 9: Arquitetura ADO.NET

2.3.2.6 LINQ to DataSet

LINQ to Dataset faz com que a busca de dados em [objetos *Dataset*] se tornem ainda mais fáceis e rápidas. Este recurso traz maior produtividade e flexibilidade para os desenvolvedores já que trabalham com [*Queries*] na sua própria linguagem de programação. LINQ to Dataset expõe, primariamente, os métodos adicionais nas classes [*DataRowExtensions*] e [*DataTableExtensions*]. Também se utiliza da arquitetura do ADO.NET 2.0, não tendo a intenção da substituir o ADO.NET 2.0 no código da aplicação. Ele permite que as consultas sejam escritas na linguagem de desenvolvimento, e fornece uma verificação de tipo tempo de compilação. Além disso, o LINQ permite que todo o poder do quadro a ser utilizado quando escrever consultas. LINQ to DataSet traz esse poder ao seu aplicativo baseado em *DataSet* (MSDN, 2010) e (GTEZINI, 2009).

2.3.2.7 Objeto

De um modo geral, deve-se encarar os objetos como sendo os objetos físicos do mundo real, tal como: carro, avião, cachorro, casa, telefone, computador, etc., por isso que às vezes é dito que orientação a objetos representa os problemas mais próximos ao mundo real, dando assim mais facilidade à programação como um todo,

mais isso não é sempre verdade, porque às vezes tem-se problemas que são extremamente funcionais.

De maneira simples, um objeto é uma entidade lógica que contêm dados e código para manipular esses dados. Os dados são denominados como sendo atributos do objeto, ou seja, a estrutura que o objeto tem, e o código que o manipula é denominado método. Um método é uma função que manipula a estrutura de dados do objeto (JACK, 2010).

2.3.2.8 LINQ to Objects

O termo "LINQ to Objects" refere-se ao uso de consultas LINQ com qualquer estrutura de banco IEnumerable ou IEnumerable<T> até mesmo coleções, diretamente, sem o uso de um intermediário provedor LINQ ou API com um sistema autônomo de tratamento de conexão. LINQ to SQL ou LINQ to XML deixa o programador preocupado apenas com o tratamento do negocio .Você pode usar LINQ para Consultas sistema autônomo List<T>, de Array, ou Dictionary<TKey, TValue>. A coleção pode ser definido pelo usuário ou pode ser retornada por .NET Framework API.

De certa forma básica, LINQ to Objects representa uma nova abordagem para coleções. Na forma antiga, havia de gravar , complexos métodos para se recuperar dados de uma coleção. Na abordagem a LINQ, você escreve código nativo LINQ e descreve o que você deseja recuperar. A figura 10 mostra LINQ To Objects.

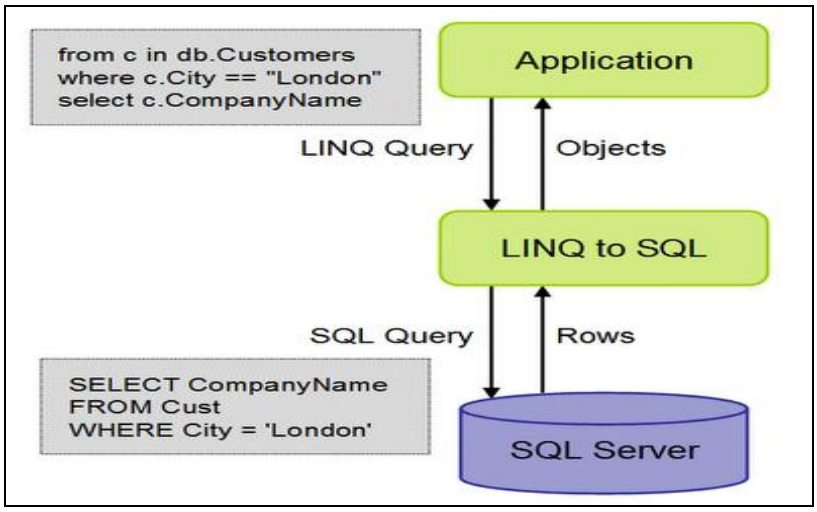

Figura 10: LINQ To Objects

2.3.2.9 XML

O XML provém de uma linguagem que a IBM inventou por volta dos anos 70. A linguagem da IBM chama-se *General Markup Language* (GML) e surgiu da necessidade que tinham na empresa de armazenar grandes quantidades de informação sobre temas diversos.

Extensible Markup Language (XML) é linguagem de marcação de dados (meta-markup language) que provê um formato para descrever dados estruturados. Isso facilita declarações mais precisas do conteúdo e resultados mais significativos de busca através de múltiplas plataformas. O XML também vai permitir o surgimento de uma nova geração de aplicações de manipulação e visualização de dados via internet.

O XML permite a definição de um número infinito de tags. Enquanto no HTML, se as *tags* podem ser usadas para definir a formatação de caracteres e parágrafos, o XML provê um sistema para criar tags para dados estruturados (JUNIOR, 2010).

Exemplo de xml

```
<?xml version="1.0" encoding="utf-8" ?>

<Clientes>
 <Cliente>
    <Nome>Macoratti</Nome>
    <Cidade>Brasilia</Cidade>
    <Idade>37</Idade>
 </Cliente>
 <Cliente>
    <Nome>Jessica</Nome>
    <Cidade>Santos</Cidade>
    <Idade>19</Idade>
 </Cliente>
 <Cliente>
    <Nome>Jefferson</Nome>
    <Cidade>Marilia</Cidade>
    <Idade>17</Idade>
 </Cliente>
 <Cliente>
    <Nome>Janice</Nome>
    <Cidade>Campinas</Cidade>
    <Idade>13</Idade>
 </Cliente>
</Clientes>
```

2.1.5.10 LINQ to XML

LINQ to XML é como o *Document Object Model* (DOM) na medida em que traz o documento XML na memória. Você pode consultar e alterar o documento e, depois de modificá-lo você pode salvá-lo em um arquivo ou criptografar e enviá-la através da Internet. No entanto, LINQ to XML difere de (DOM) por que. Proporciona um novo modelo de objeto que é mais leve e mais fácil de trabalhar, por ter as vantagens vindas dos avanços na linguagem Visual C # 2008.

A vantagem mais importante do LINQ to XML é sua integração com *Language-Integrated Query* (LINQ). Esta integração permite que você escreva consultas sobre o documento XML na memória para recuperar coleções de elementos e atributos. A capacidade de consulta do LINQ to XML é comparável em termos de funcionalidade (embora não na sintaxe) com XPath e XQuery. A integração do LINQ no Visual C #

2008 proporciona uma simultaneidade na digitação, verificação em tempo de compilação, depurador e suporte melhorado.

Outra vantagem do LINQ to XML é a capacidade de utilizar os resultados da consulta como parâmetros para XElement e XAttribute construtores de objetos permitem uma abordagem poderosa para criar árvores XML. Esta abordagem chama-se construção funcional, permite que os desenvolvedores facilmente transformem árvores XML de um formato para outro (MACORATTI, 2010).

Exemplo de LINQ to XML

```
protected void carregaDLL()
    {
        XmlDocument doc = new XmlDocument();
        doc.Load(Server.MapPath("App_Data/Clientes.xml"));
        XmlNodeList nodeList = doc.SelectNodes("Clientes/Cliente");

        foreach (XmlNode node in nodeList)
            ddlCidade.Items.Add(new
ListItem(node.SelectSingleNode("Cidade").InnerText));
    }
```

2.4 IMPORTÂNCIA DO LINQ

2.4.1 LINQ Atualmente

Como explanado a pouco existe uma “guerra de mercado” entre as tecnologias em razão desta “guerra” que nasceram grande parte das evoluções tecnológicas existentes, com a tecnologia LINQ não foi diferente embora, tenha entrado para o .net *framework* como uma tecnologia nova existe um suave tom de resposta aos concorrentes da Microsoft uma vez que LINQ divide espaço com NHibernate, Hibernate acredita-se ser os dois concorrentes diretos de LINQ. O NHibernate até com mais peso nesta concorrência sendo que o mesmo foi criado para ser utilizado

em aplicações .NET e por isso pode se dizer que o LINQ briga diretamente com o NHibernate,no que se pode chamar disputa interna pois as duas tecnologias disputam literalmente o mesmo espaço, pois atuam em amparo ao mesmo produto que é a tecnologia .NET . Já Hibernate briga diretamente pelo mercado com LINQ, pois disputam o mercado externo ou de vertentes dos que usam Java e por fim utilizam o HIBERNATE e dos que usam o LINQ que como já abordado faz parte do pacote de melhorias da ferramenta .NET .

2.4.2 LINQ Tecnologia

A tecnologia LINQ portou muito do padrão de programação das ferramentas Microsoft no que diz respeito à facilidade em se programar, utilizando de tecnologias sensitivas ou como queiram automáticas, mas o principal do LINQ é dar suporte a plataforma .NET como um produto nativo pois nota se, com a chegada do LINQ e tantas outras tecnologia que a intenção das invenções da Microsoft muitas vezes são a penas para mostrar sua auto suficiência. É claro que LINQ tem um papel importante dentro da plataforma .NET pois um produto nativo em regra tem uma vantagem em relação a produtos acoplados tanto em desempenho quanto suporte .Isso é ainda mais aparente quando se tratando de .NET onde cada vez mais se busca a automatização de processos, a finalidade da Microsoft fica mais explicita quando notamos na padronização do estilo de programar que a empresa difunde onde se bem tratadas as bases dos dados e bem declaradas as ações a serem efetuadas pelo aplicativo, coisas como configuração de conexão e criação de classes ficam quase em segundo plano, deixando os programadores apenas focados com a programação em si.

2.4.3 Pontos Positivos Do LINQ

LINQ tem a seu favor uma grande comunidade de usuários o que nos dias de hoje pode ser considerado o ibope das tecnologias, ou seja, é o que as mantém no mercado. Faz parte da plataforma .NET testada e aprovada no mercado mundial como marca consistente e de tecnologia durável. LINQ tem dado resultados significativos em relação a suporte a plataforma .NET, esta tecnologia suporta muito bem a interatividade do estilo de programação das ferramentas Microsoft. LINQ consegue estar perto de seus concorrentes em relação a desempenho se comparados, espera se muito desta tecnologia, pois do dia de sua criação até os dias de hoje atualizações foram constantes para o aprimoramento da tecnologia.

2.4.4 Pontos Negativos Do LINQ

LINQ é uma ferramenta comercial não aberta como Hibernate ou NHibernate, ainda sofre muita especulação sobre sua sobrevivência ou longevidade visto seu tempo de existência ,embora tenha nascia com o intuito de dar suporte a todos os bancos de dados poucos avanços foram feitos em relação a múltiplos drives dificultando a utilização com outros bancos que não os SQL Server, a tecnologia tem uma dura disputa de mercado com Hibernate e NHibernate pois fazer essa transição ainda é de risco pois seria necessário mudar estruturas ou arquiteturas inteiras.

2.5 NHibernate

NHibernate é uma parte do *Hibernate Core Java* para o .NET *Framework*. Ele lida com persistência simples .NET par um banco de dados relacional subjacente. Dada uma descrição XML de suas entidades e relacionamentos, NHibernate gera automaticamente o SQL para carregar e armazenar objetos. Opcionalmente, pode-se descrever o mapeamento de Metadados com atributos em seu código fonte.

NHibernate suporta persistência transparente, aulas de seu objeto não tem que seguir um modelo de programação restritiva. As classes persistentes não precisam implementar nenhuma interface ou herdar de uma classe base especial. Isto torna possível projetar a lógica de negócios simplesmente .NET (CLR), objetos e linguagem orientada a objetos.

Originalmente sendo uma parte do Hibernate 2.1, a API do NHibernate é muito semelhante ao do Hibernate. Todo o conhecimento do Hibernate e da documentação do Hibernate existente é, portanto, diretamente aplicável ao NHibernate (NHFORGE, 2010). A figura 11 mostra a arquitetura NHIBERNATE.

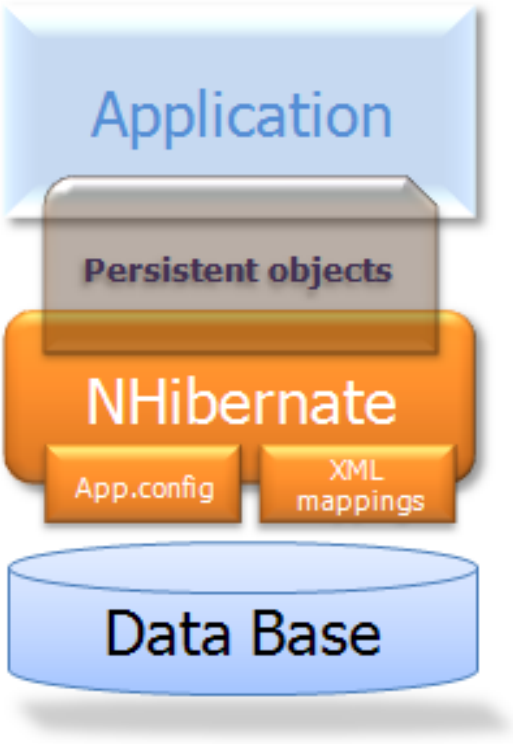

FIGURA 11: ARQUITETURA NHIBERNATE

Esse diagrama mostra o NHibernate usando a configuração de dados e banco de dados para prestação de serviços de persistência (e objetos persistentes) para o aplicativo.

2.6 Hibernate

Hibernate foi iniciado em 2001 por Gavin King como uma alternativa ao uso de beans de entidade EJB2. Sua missão na época era simplesmente oferecer melhores recursos de persistência que o oferecido por EJB2 simplificando as complexidades e permitindo recursos ausentes.

Historicamente, o Hibernate facilitou o armazenamento e recuperação de objetos de domínio Java através de mapeamento Objeto/Relacional. Hoje, o Hibernate é uma coleção de projetos relacionados permitindo que os desenvolvedores a utilizar o estilo de domínio modelos-POJO em suas aplicações de forma que se estende bem além do mapeamento Objeto/Relacional. A figura 12 mostra HIBERNATE.

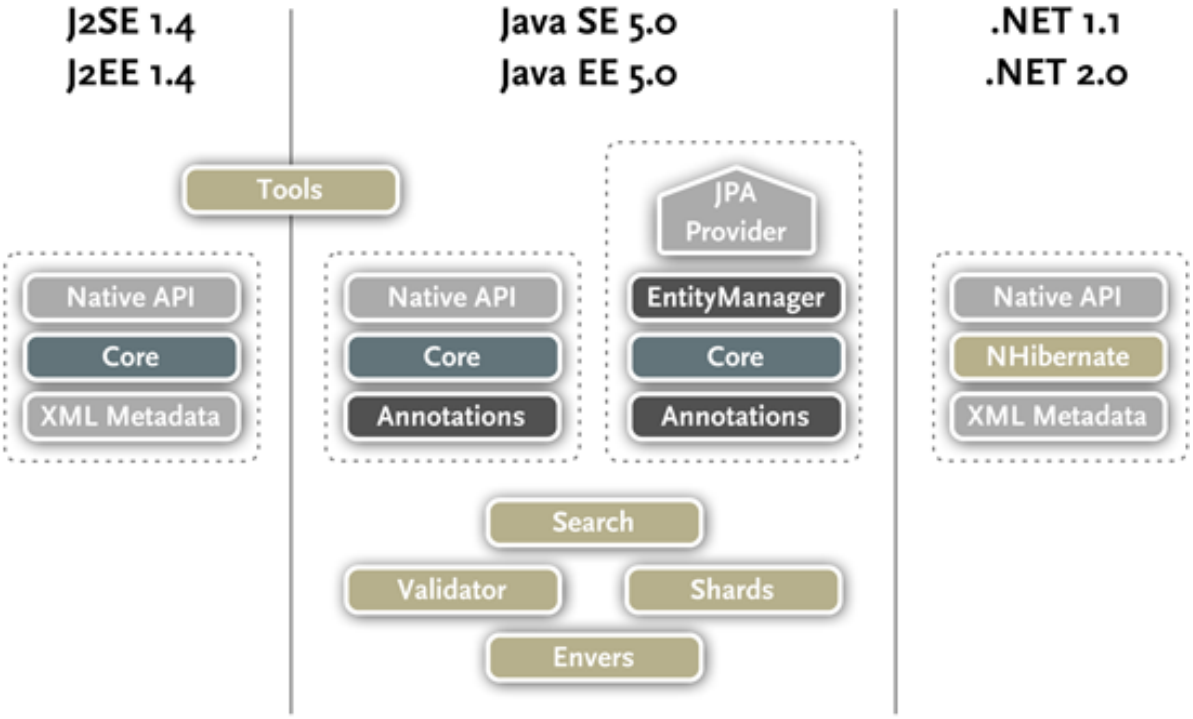

FIGURA 12: HIBERNATE

3. DESENVOLVIMENTO DO APLICATIVO

Neste capítulo será apresentada a modelagem do problema, onde foram divididos por módulos para facilitar o entendimento das fases a serem desenvolvidas. Para a realização deste trabalho foram executados procedimentos de especificação e implementação visando o cumprimento dos objetivos.

3.1 Descrição do Problema

Neste trabalho será desenvolvido um aplicativo utilizando as tecnologias C# e LINQ, para gerenciamento de uma loja de perfume. Este gerenciamento está relacionado com os dados cadastrais de clientes e produtos. Será utilizando a Linguagem C# para desenvolver este aplicativo por ser parte integrante do pacote .NET *FRAMEWORK*, pois o propósito principal deste aplicativo é reunir conhecimento para implantar a tecnologia LINQ.

3.2 Modelagem do Problema

A modelagem é de suma importância, pois ela serve para dar uma idéia geral do problema que será abordado. A figura 13 mostra a modelagem do problema, e ela foi dividida em três etapas para facilitar o desenvolvimento de cada parte que constitui o aplicativo.

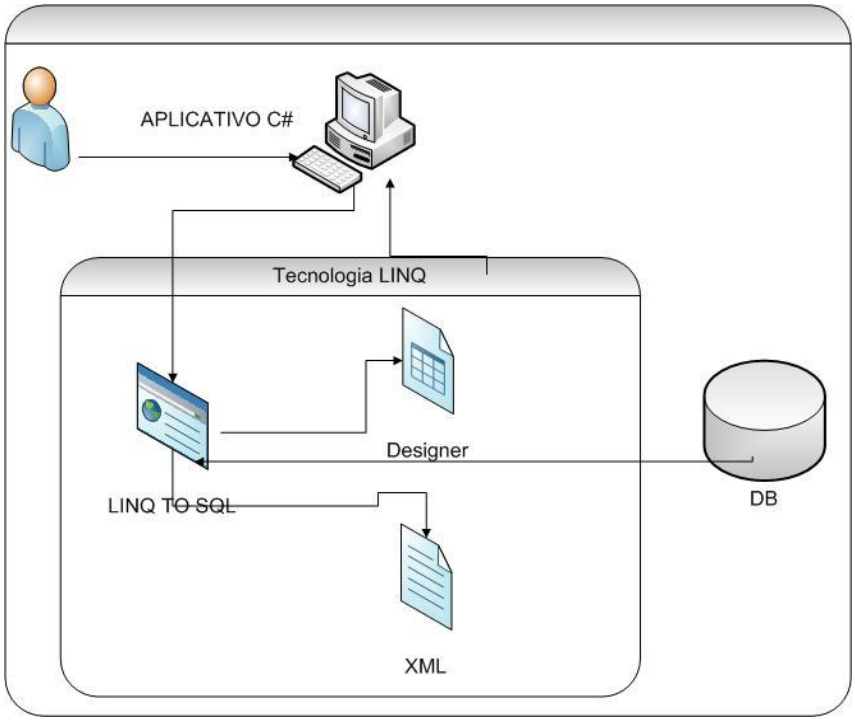

Figura 13: Modelagem do problema.

Etapa 1 - Criação do Banco de dados.

Nesta etapa será criado o Banco de Dados SQL *Server* 2005, pois apresenta melhor resposta em referência a outros, oferecendo melhor suporte a tecnologia LINQ.

Etapa 2 - Criação do Sistema e Interface do Usuário

Nesta etapa será desenvolvido um aplicativo *desktop*, com a ferramenta *Visual Studio 2008*, onde os dados enviados serão recebidos através do Windows Form, fazendo o processamento das informações e efetuando o cadastro dos dados no banco. O acesso as informações do aplicativo pode ser obtidas por qualquer usuário. A interface de comunicação do usuário com os dados será desenvolvida na linguagem C#.

Etapa 3 - Implementação da tecnologia LINQ.

Nesta etapa será desenvolvido o mapeamento LINQ-TO-SQL para a criação das classes e métodos seguindo a formatação das tabelas, responsável em fazer a integração do aplicativo com o banco por meio de conexão XML.

3.3 Especificação

Para fazer a especificação deste aplicativo foi utilizada uma metodologia orientada a objetos, representada em diagramas UML (*Unified Modeling Language*), utilizando como ferramenta a Microsoft Office Visio e *Enterprise Architect*. O primeiro diagrama utilizado na especificação é o de casos de uso, seguido pelo diagrama de classes, diagrama de atividades e por último os diagramas de seqüência especificando o funcionamento do aplicativo.

3.3.1 Diagrama de casos de uso

O aplicativo possui seis casos de uso, como mostra a figura 14.

- **Manter Clientes:** responsável por todas as operações relacionadas ao cliente.
- **Manter Cataloga de Produtos:** responsável por todas as operações relacionadas aos produtos.
- **Incluir:** responsável por armazenar objetos novos no banco de dados.
- **Pesquisar:** responsável por recuperar os objetos armazenados no banco de dados.
- **Excluir:** responsável por excluir o objeto anteriormente pesquisado.
- **Atualizar:** responsável por atualizar o objeto anteriormente pesquisado.
- **Visualizar Catalogo:** responsável pelas navegação fotos cadastrais dos produtos

- **Orçamento:** responsável em orçar o valor do produto e demonstrar as formas de pagamento.
- **Alterar valor do dólar:** responsável por adequar cambial diário do dólar.

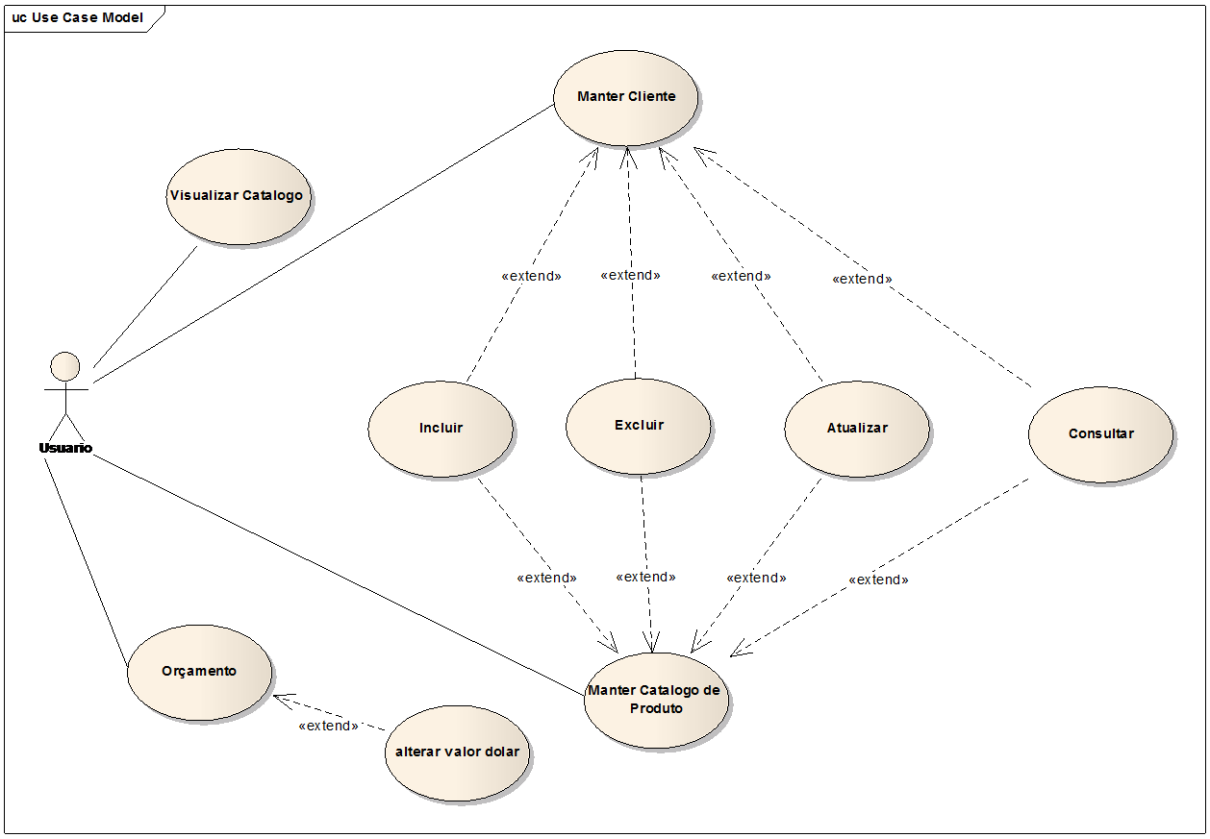

Figura 14: Casos de uso.

3.3.2 Diagramas de classes

As classes utilizadas no desenvolvimento deste aplicativo foram separadas em pacotes, sendo eles:

- *DAL* embora tenha sido criado o pacote neste aplicativo, o mesmo não foi utilizado, pois as necessidades das informações foram locais devido a quantidade pequena de dados. A figura 15 mostra o diagrama de classe do pacote *DAL*.

- LINQ TO SQL uma vez representado o banco na aplicação uma conexão *default* é configurada responsável.

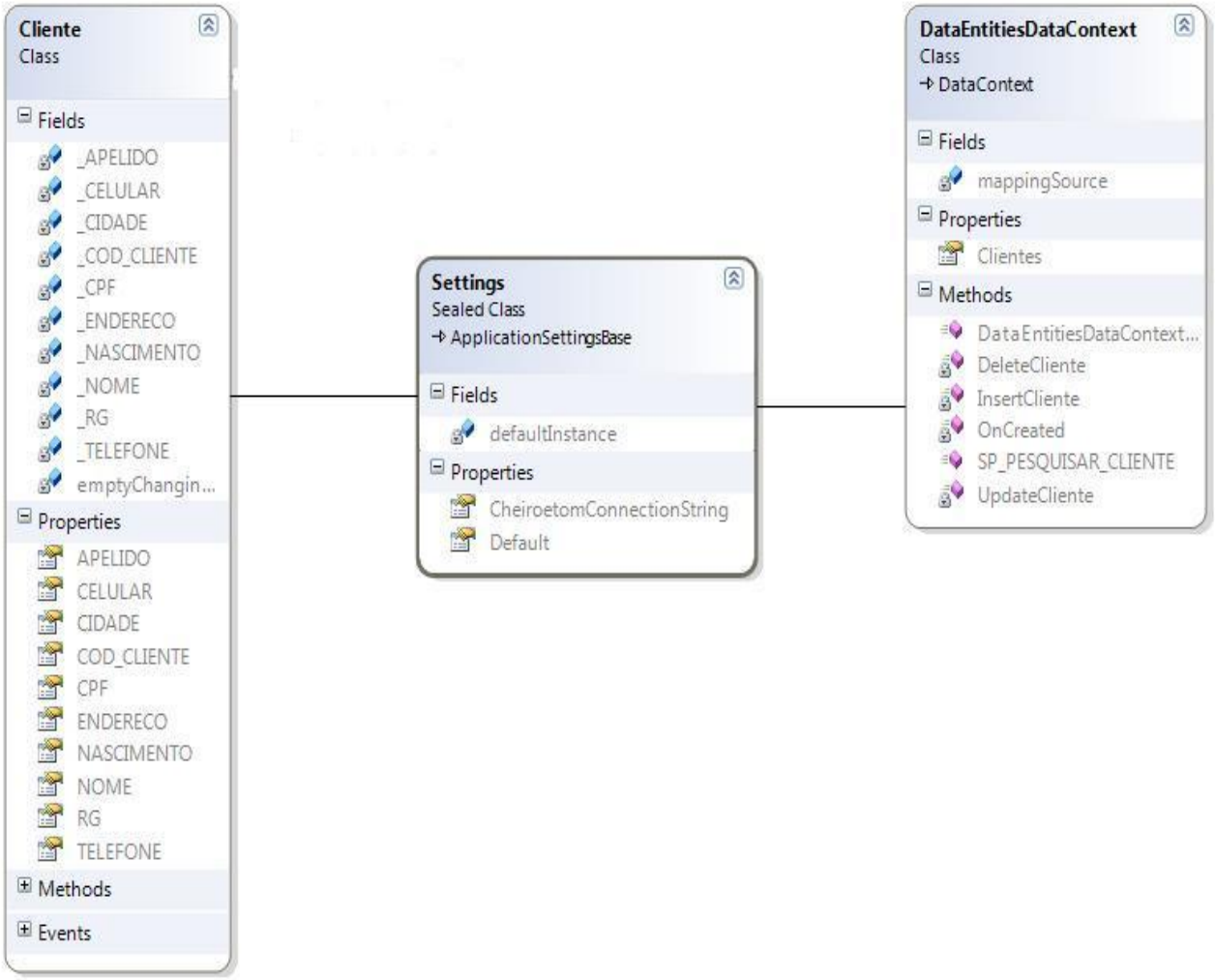

Figura 15: Diagrama De Classes Pacote Dal

- BAL, a classe BLL (que chamamos de BAL) torna transparente a camada de acesso a dados - DAL através do mapeamento O/R realizado via LINQ To SQL sendo assim responsável por retornar os resultados das operações solicitadas via interface
- Interface responsável pela iteração do usuário com o aplicativo.

A figura 16 mostra Diagrama de Classes Interface.

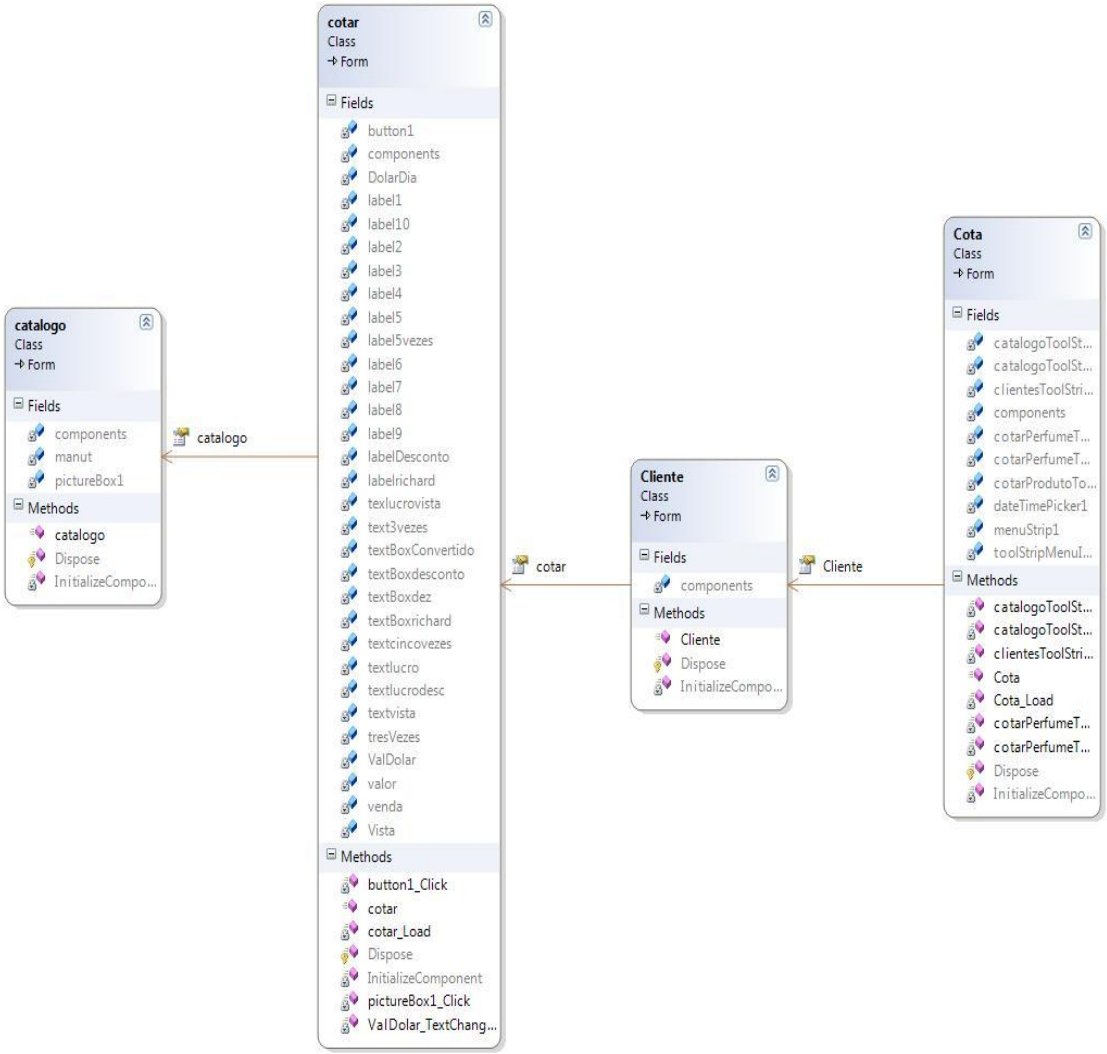

Figura 16: Diagrama de Classes Interface

3.3.3 Diagrama de atividades

O diagrama de atividades visa representar o controle de fluxo das informações entre as atividades de um sistema. Na figura 17, é apresentado o diagrama de atividades do aplicativo, destacando todos os procedimentos necessários para o cadastro de clientes, catalogo de produtos, consulta e orçamento.

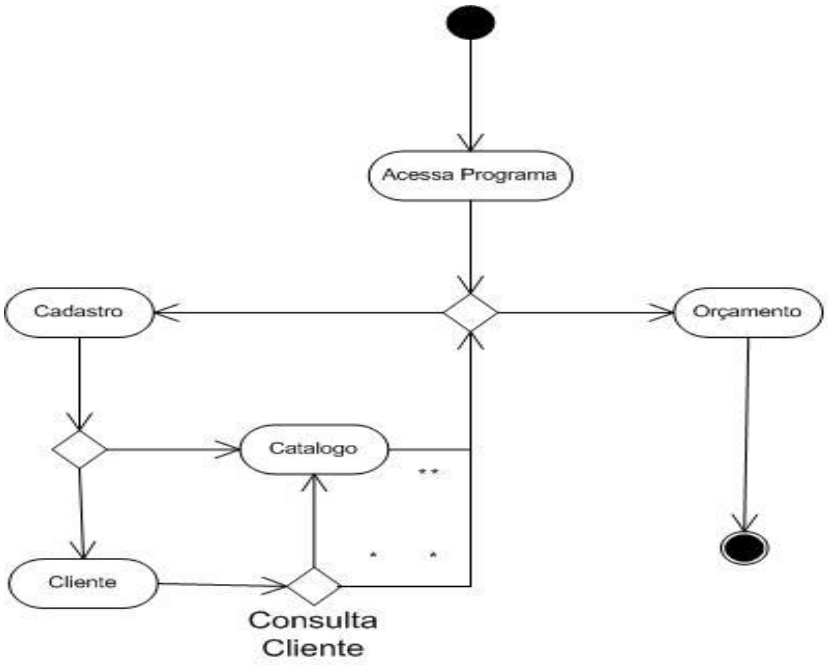

Figura 17: Diagrama de atividades.

3.3.4 Diagramas de seqüência

Os diagramas de seqüência representam a seqüência das ações ocorridas em um conjunto de classes, demonstrando como ocorre a troca de mensagens entre elas. Para cada caso de uso especificado, há um diagrama de seqüência, conforme detalhamento a seguir.

3.3.4.1 Manter clientes

Este diagrama de seqüência representado na figura 18 mostra as ações que podem ser executadas neste caso de uso, tais como: inserir, pesquisar e excluir cliente.

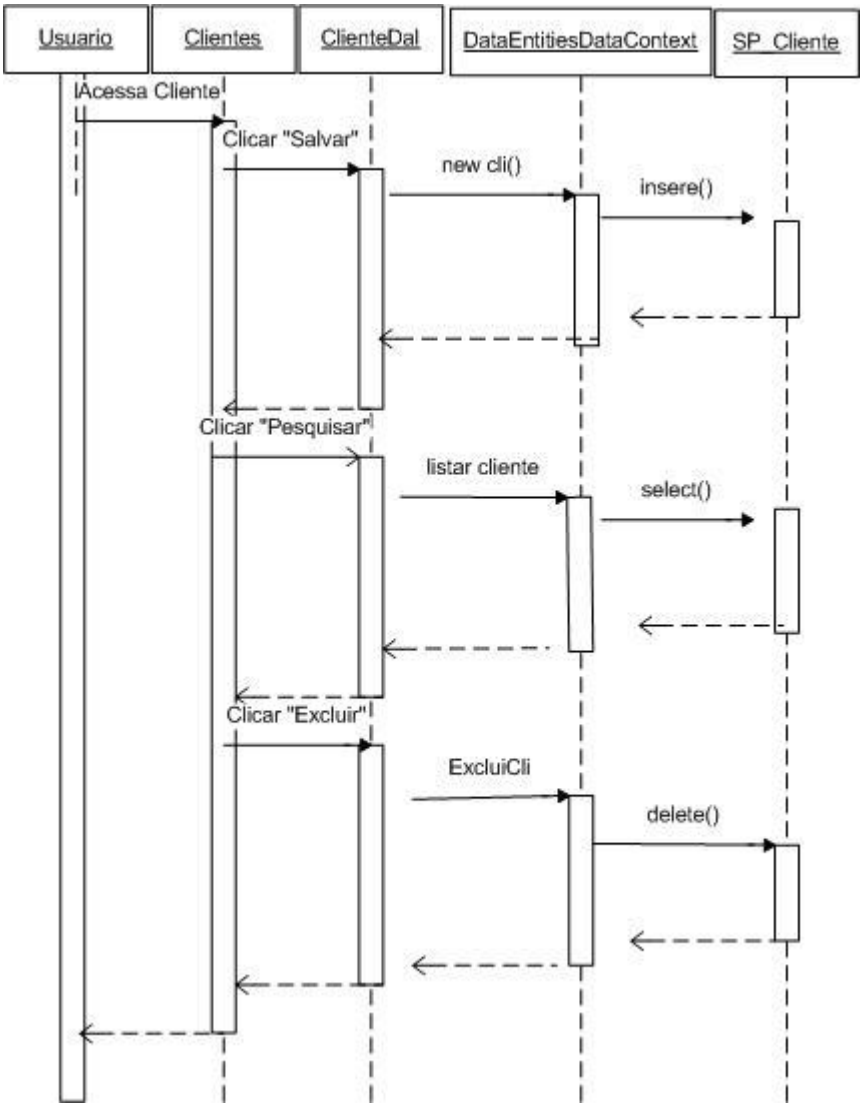

Figura 18: Diagrama de seqüência – Manter clientes

3.3.4.2 Manter Catalogo de Produto

O diagrama de seqüência que está representado na figura 19 mostra as ações que podem ser executadas neste caso de uso, tais como: inserir, pesquisar, alterar e excluir produtos.

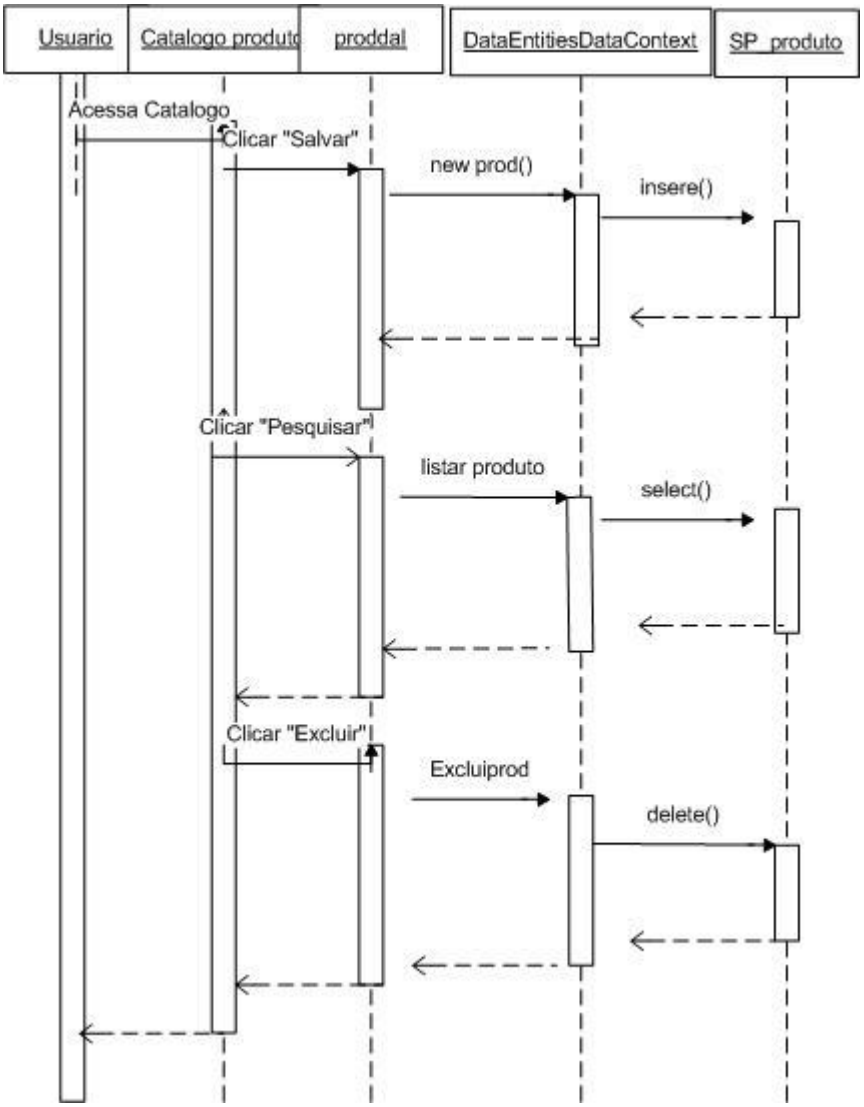

Figura 19: Diagrama de seqüência – Manter Catalogo de produto.

3.3.4.3 Visualizar orçamento

Este diagrama é executado quando o usuário entra na página de orçamento dos produtos, ele é responsável em calcular o valor do produto no dia, pois o orçamento depende do valor cambial para se obter um valor final do produto e demonstrar formas de pagamento. A figura 20 mostra Diagrama de seqüência – Visualizar orçamento.

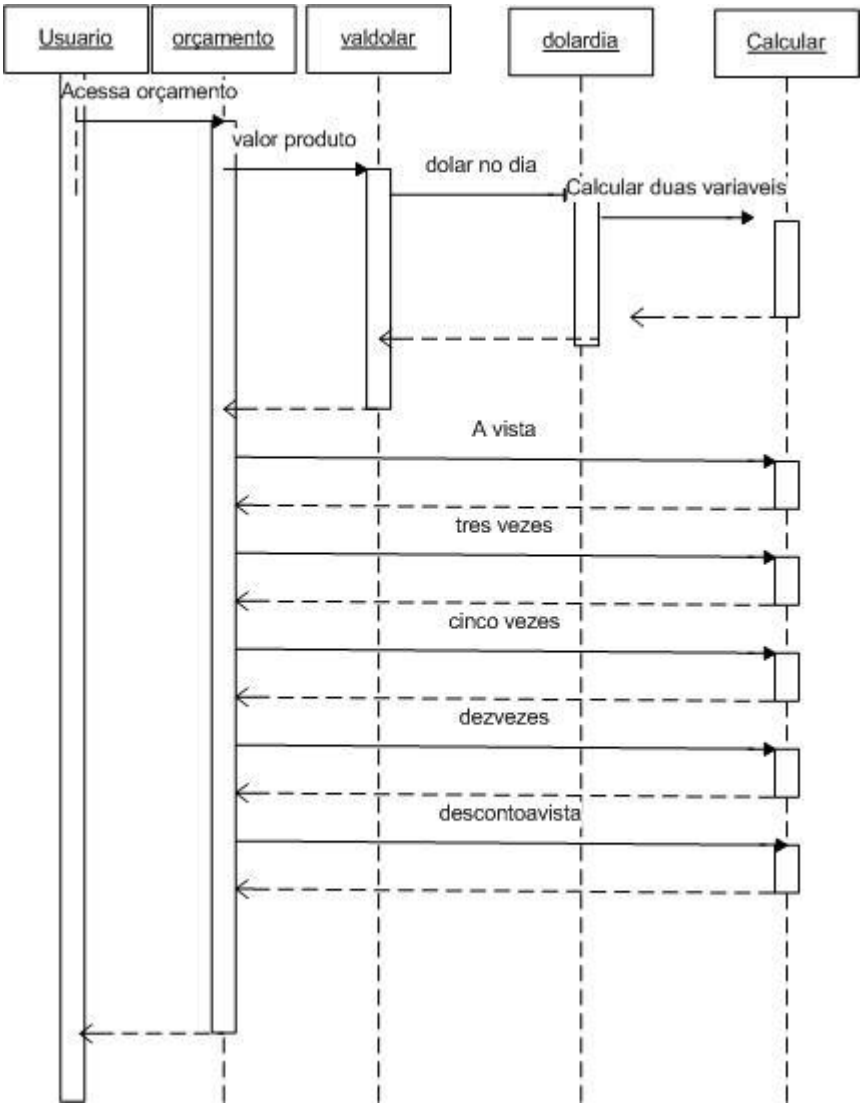

Figura 20: Diagrama de seqüência – Visualizar orçamento.

3.4 Implementação do Aplicativo

Serão apresentados os tópicos referentes à implementação do aplicativo desenvolvido neste trabalho, utilizando o ambiente de programação *Visual Studio2008*

3.4.1 Operacionalidade da implementação

O projeto desenvolvido neste trabalho é uma aplicação *desktop* com tecnologia LINQ para tratar toda área de conexão do aplicativo com o banco de dados, capaz de cadastrar clientes, catalogo de produtos e disponibilizar o orçamento de um produto.

A figura 21 apresenta a interface principal do aplicativo, onde o conteúdo da mesma pode ser dividida nas implementações dos casos de uso que seguem abaixo.

Figura 21: Pagina inicial do aplicativo *Cheiro&Tom*

3.4.1.1 – Implementação do caso de uso “Manter clientes”

O caso de uso “Manter clientes” ocorre quando um usuário deseja incluir, pesquisar, alterar ou excluir um cliente no aplicativo.

Para a inclusão de um cliente é necessário o preenchimento dos campos, sendo obrigatórios somente os campos: código, nome, endereço, cidade, RG e CPF. Após estes passos, basta confirmar a inclusão através do botão “Salvar”.

A figura 22 mostra Selecionando um formulário para incluir um cliente

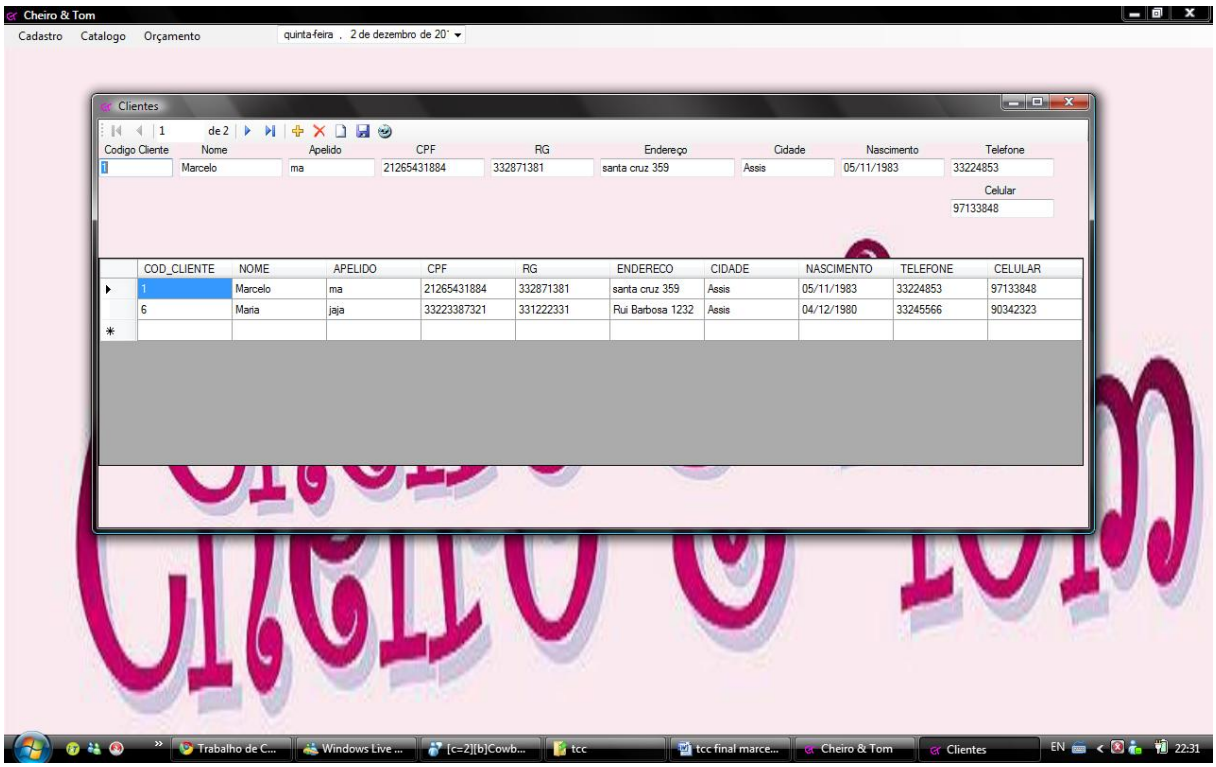

Figura 22: Selecionando um formulário para incluir um cliente.

Para incluir, excluir ou alterar um cliente, primeiro é necessário clicar no ícone cliente, para pesquisar o cliente desejado. Após a escolha do mesmo, será carregado todas as suas informações nos campos correspondentes, agora basta escolher entre alterar os dados ou excluí-lo através dos botões “Atualizar” ou “Excluir”.

3.4.1.2 Implementação do caso de uso “Manter catalogo de produto”

O caso de uso “Manter catalogo de produtos” ocorre quando um usuário deseja incluir, pesquisar, alterar ou excluir um produto no aplicativo.

Para a inclusão de um produto é necessário o preenchimento dos campos, sendo obrigatórios somente os campos: código, nome, tipo, valor de custo e foto.

Para excluir ou alterar um produto, primeiro é necessário clicar no ícone catalogo. Após a escolha do mesmo, será carregado todas as suas informações nos campos correspondentes, agora basta escolher entre alterar os dados ou excluí-lo através dos botões “Atualizar” ou “Excluir”.

A figura 23 a interface de comunicação para a inclusão de produto.

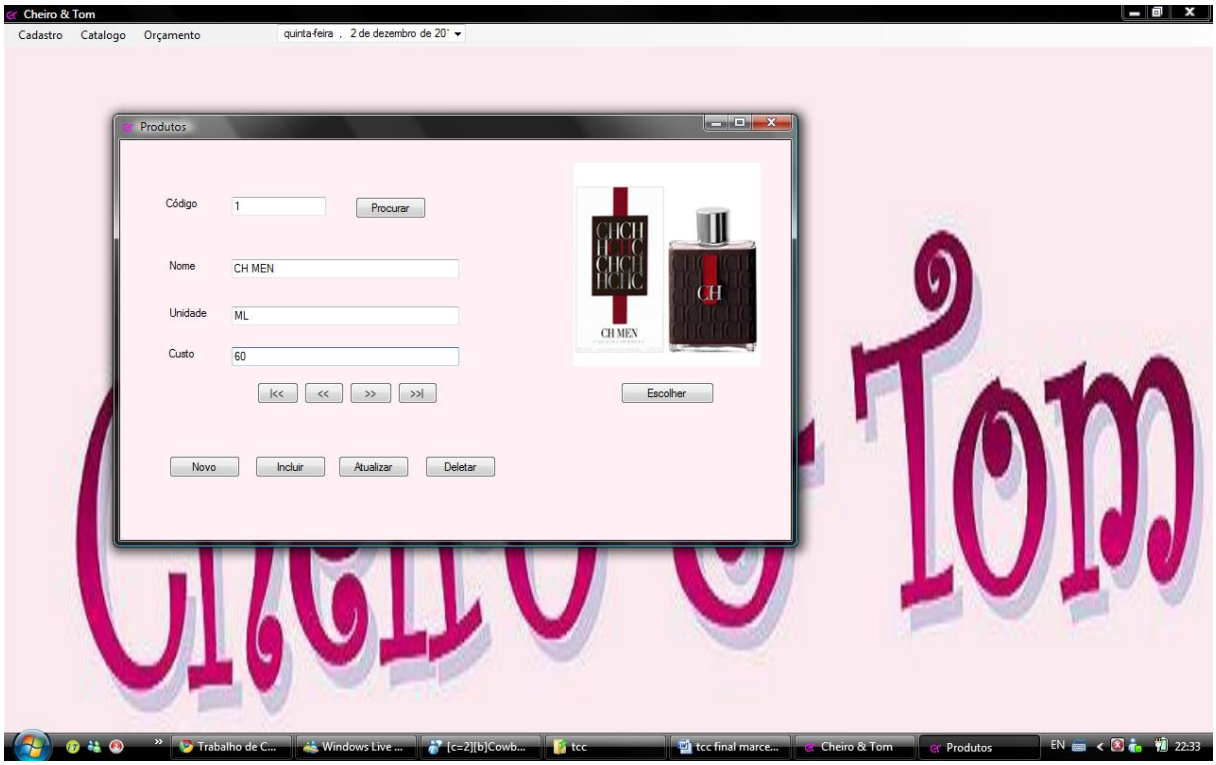

Figura 23: Página para a inclusão de produto

3.4.1.3 Implementação do caso de uso “Orçamento”

O caso de uso “Orçamento” ocorre quando um usuário clica no ícone “Orçamento” da página inicial. Neste momento a abre se uma pagina onde o usuario encere o valor do produto em seguida o valor do dólar diário ao clicar em calcular o mesmo terá o valor do produto convertido, levando em consideração o cambio o valor com transporte e impostos, valor venda a vista, valor do seguro caso haja, valor parcelado de 3 a 10x e valor de suas parcelas e demonstrativo de lucro em cada tipo de venda.

A figura 24 mostra a interface de Orçamento.

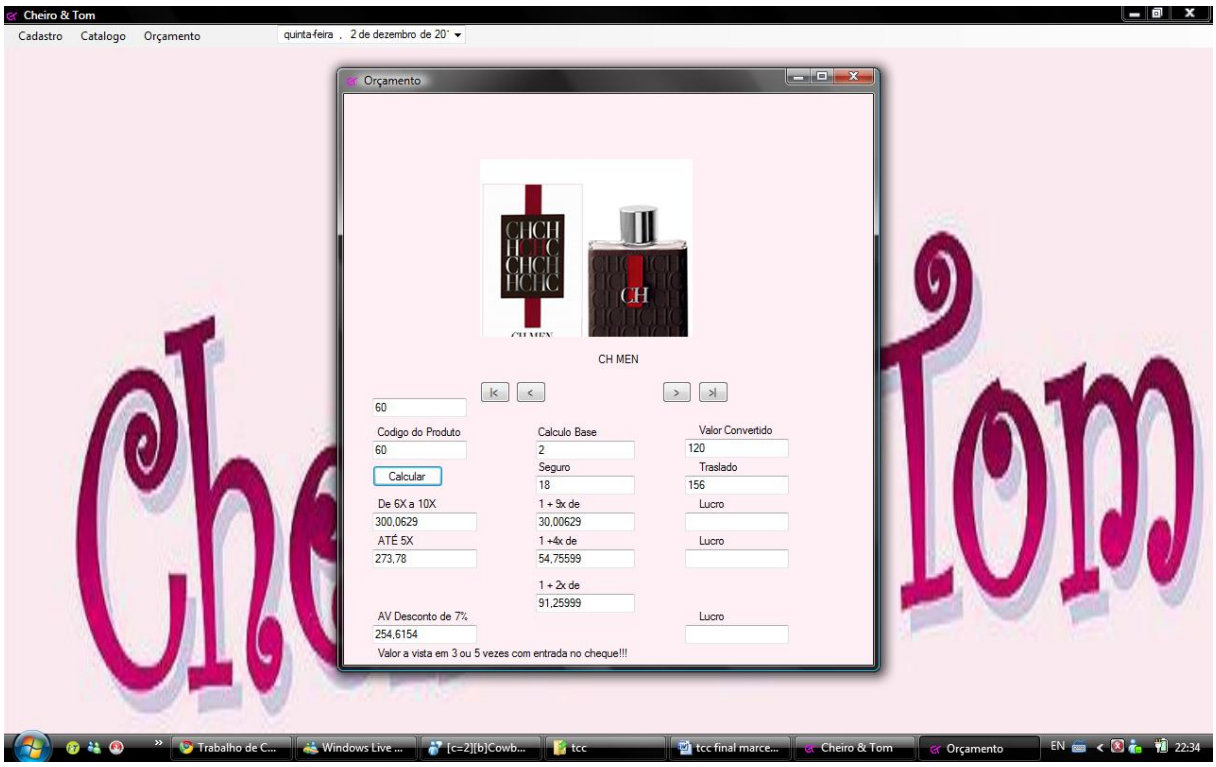

Figura 24: Página de Orçamento.

3.4.1.4 Implementação do caso de uso “Visualizar Catalogo”

O caso de uso “Visualizar Catalogo” ocorre quando um usuário clica no ícone “Catalogo” da página inicial. Neste momento abre uma pagina onde o usuário pode navegar por todo catalogo de fotos dos produtos existentes no aplicativo que estejam cadastrados.

A figura 25 mostra o Catalogo De Fotos.

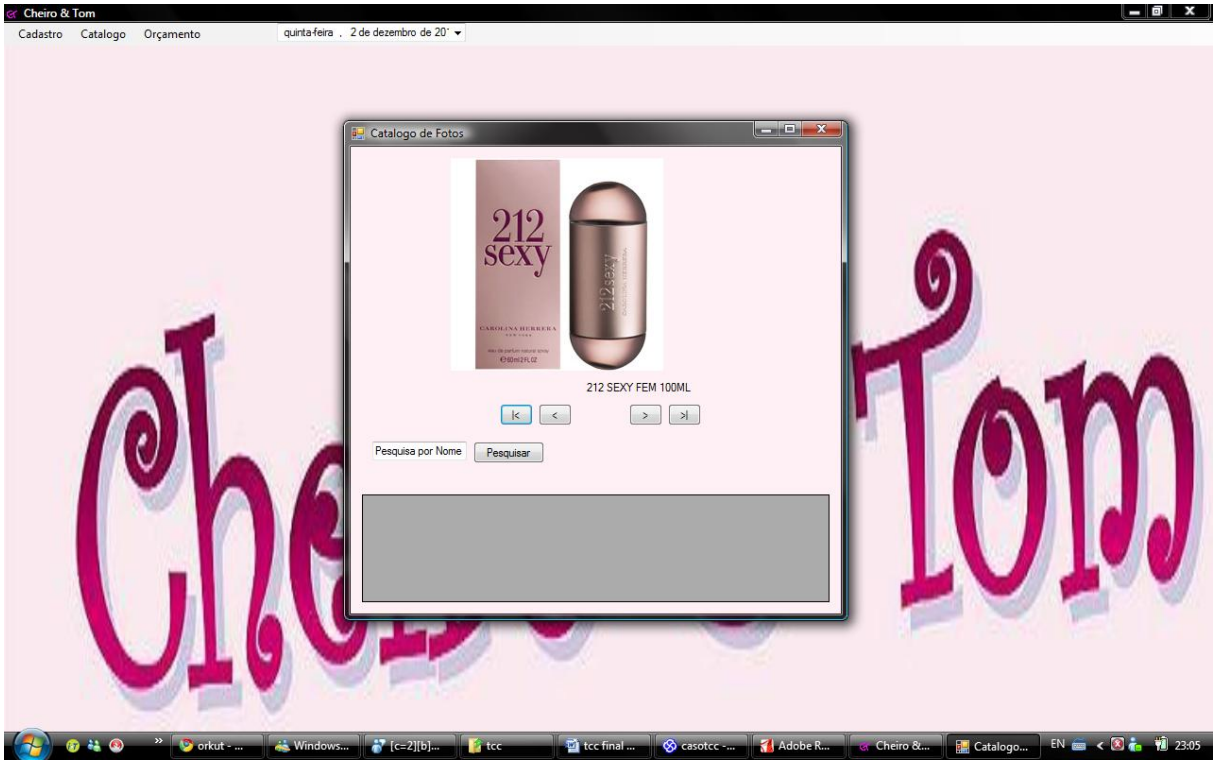

Figura 25: Catalogo De Fotos.

CONCLUSÃO

Este trabalho de conclusão de curso foi motivado pelo crescente interesse na utilização de conceitos de programação ágeis e interativas onde a facilidade das integrações dos aplicativos com bancos de dados sejam tratados como uma simples ação e não como parte critica de um programa. O grande desafio foi aprender a desenvolver um aplicativo usando um novo conceito de programação onde classes são representação do banco e métodos são replicas de stored procedures .

Outro desafio foi aprender todo o mecanismo que envolve o princípio básico da linguagem C# a utilização da ferramenta Visual Studio 2008.

As pesquisas realizadas sobre as ferramentas, *frameworks*, tecnologias e arquiteturas envolvidas para o desenvolvimento do aplicativo, são as principais contribuições deste trabalho. Contribui de forma direta para os alunos, como material de estudo e para empresas que desejam adotar o uso destes padrões de projeto em suas aplicações, melhorando a qualidade e a produtividade. Alem disso, os conhecimentos adquiridos para a realização deste trabalho foram de suma importância para o crescimento profissional e acadêmico. Sem dúvida esses conhecimentos serão fundamentais para a elaboração de outros projetos de pesquisa no futuro.

A aplicação *Desktop* se mostrou de grande serventia na contribuição do gerenciamento de informações cadastrais tanto de produtos como de clientes, seu modulo orçamento dinamiza todo processo de cálculos antes feitos manualmente.

Como extensão deste trabalho pode-se estudar técnicas para que se faça um produto com todos os módulos representando uma loja no aplicativo e com isso fazer um sistema completo.

Outra sugestão seria o estudo *Web* para transformar o aplicativo em um sistema Web onde ele poderia ser acessado de qualquer lugar transformando uma loja em franquia e o mesmo ser adotado por todas as lojas.

REFERÊNCIAS

ALVAREZ, M. A.: **Objetivos e usos do XML**. Disponível em: <http://www.criarweb.com/artigos/431.php>. Acesso em junho de 2010.

AMSDESIGN.: **B2C – Business to Consumer**. Disponível em: <http://www.amsdesign.com.br/B2C_Business_to_Consumer.asp>. Acesso em junho de 2010.

BARALE, R. F.: **Desenvolvimento de um sistema de Vendas na web**. Uberlândia, 2007.

BAUER, C. e KING, G.: **Hibernate in Action**, Manning Publications, 2004.
BERGSTEN, H.: **JavaServer Faces**, O'Reilly, 2004.

BOX,D; HEJLSBERG, A: **O PROJETO LINQ**. Disponível em http://msdn.microsoft.com/pt-br/library/bb308959.aspx . Acesso em junho de 2010.

CAMARGO, DALTON: **QUAERE-LINQ PARA JAVA**, JAVAFREE.ORG. Disponível em http://javafree.uol.com.br/noticia/3647/Quaere-LINQ-para-Java.html . Acesso em :Março de 2010[5]

CENTER,**.**NET Framework Developer**: APRENDER LINQ**, Disponível em http://msdn.microsoft.com/en-us/netframework/aa904594.aspx. Acesso em: Fevereiro de 2010.

DURÃES, RAMON: **INTRODUÇÃO LINQ TO SQL** , Disponível em http://imasters.uol.com.br/artigo/7156/bancodedados/introducao_linq_to_sql/ . Acesso em junho de 2010.

EDUCACAO ,PORTAL: **História e características da linguagem C#.** Disponível em:<http://www.portaleducacao.com.br/informatica/artigos/6137/historia-e-caracteristicas-da-linguagem-c> . Acesso em março de 2010.

FACTS, IT: **Global RDBMS market**. Disponível em http://www.itfacts.biz/global-rdbms-market-oracle-443-ibm-21-microsoft-158/10857. Acesso em: fevereiro de 2010.[2]

GTEZINI,.NET: **.NET Framework 3.5 - LINQ to DataSet .Disponivel em** http://gtezini.blogspot.com/2009/02/net-framework-35-linq-to-dataset.html .Acesso em outubro de 2010.

HADDAD, Renato Ibrahim**. LINQ e C# 3.0 A Solução em consultas para desenvolvedores.** 1ª edição – São Paulo: Editora Érica LTDA., 2009.[3]
STELLMAN, Andrew; GREENE, Jennifer**.Use a Cabeça C#.** 1ª edição – São Paulo: Editora ALTA BOOKS, 2008.

JACK,eti.br: **Programação Orientada A Objetos .** informações e conteúdo sobre T.I Disponível em http://www.jack.eti.br/www/arquivos/apostilas/java/poo.pdf . Acesso em junho de 2010.[11]

JUNIOR, Miguel Benedito Furtado . **XML - Extensible Markup Language** . Curso: redes de computadores. Prof. Otto Carlos Muniz Bandeira Duarte . Centro de tecnologia Eng. Eletrônica .Universidade Federal Do Rio De Janeiro.Disponivel em http://www.gta.ufrj.br/grad/00_1/miguel/link1.htm acesso em junho de 2010.

LAI,ERIC: **Oracle prorrogada vantagem sobre a IBM, em 2006, dados de mercado**, COMPUWORD , Disponível em http://www.computerworld.com/s/article/9017898/IDC_Oracle_extended_lead_over_IBM_in_2006_database_market?intsrc=news_list. Acesso em: fevereiro de 2010.

MACORATTI, José, Carlos: **Filtrando e exibindo dados XML com LINQ**, Disponível em http://imasters.uol.com.br/artigo/12697/aspnet/asp_net_filtrando_e_exibindo_dados_xml_com_linq/ . Acesso em junho de 2010.

MACORATTI, José Carlos**. Apresentando LINQ to Entities (Entity Framework)** autor dos livros "Aprenda Rápido: ASP" e "ASP, ADO e Banco de Dados na Internet". Mantenedor do site macoratti.net.

http://imasters.uol.com.br/artigo/10457/dotnet/apresentando_linq_to_entities_entity_framework/ . Acesso em março de 2010.[10]

MSDN,BIBLIOTECA:**Expressões de consulta LINQ**. Disponível em http://msdn.microsoft.com/pt-br/library/bb397676(v=VS.90).aspx. Acesso em :Março de 2010.[4]

MSDN: **Acesso a Dados e Storage**. BIBLIOTECA. Disponível em http://msdn.microsoft.com/pt-br/data/cc299380(en-us).aspx . Acesso em junho 2010.[9]

MSDN: **Introdução à linguagem C# e ao Framework .NET** . BIBLIOTECA .Disponivel em http://msdn.microsoft.com/pt-br/library/z1zx9t92(v=VS.90).aspx .Acesso em junho de 2010.

NHFORGE,.ORG : **NHibernate Documentação de Referência** .Disponivel em http://nhforge.org/doc/nh/en/index.html#performance-cache .Acesso em outubro de 2010.

SALES,Juliano: **Introdução ao C#** . CODIFICANDO.NET Disponível em :< http://comunidade.codificando.net/profiles/blogs/introducao-ao-c> Acesso em março de 2010.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. **Sistemas de Banco de Dados.** Tradução de Daniel Vieira. 5ª edição – Rio de Janeiro: Elsevier, 2006.

TEOREY, Toby; LIGHTSTONE, Sam; NADEAU, Tom. **Projeto e Modelagem de Banco de Dados.** 4. ed. Tradução de Daniel Vieira. Rio de Janeiro: Editora Elsevier, 2007.

WARREN, Matt: **A ORIGEM DO LINQ TO SQL**. Disponível em http://blogs.msdn.com/b/mattwar/archive/2007/05/31/the-origin-of-linq-to-sql.aspx . Acesso em: junho de 2010.[6]

ZEMEL, TÁRCIO: **O que é um framework.** CodeIgniter Brasil .Disponivel em http://codeigniterbrasil.com/passos-iniciais/o-que-e-um-framework-definicao-e-beneficios-de-se-usar-frameworks/ .Acesso em julho de 2010.

W3SCHOOLS.: **XML Tutorial**. Disponível em: <http://www.w3schools.com/xml/>. Acesso em junho de 2010.

XMLDESIGNER.: **Backgrounder da Tecnologia XML**, Disponível em: <http://msdn.microsoft.com/pt-br/library/8ktfywf4(VS.80).aspx>. Acesso em junho de 2010.