UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO – BACHARELADO

PROTÓTIPO DE SISTEMA CRM PARA DISPOSITIVOS MÓVEIS UTILIZANDO A TECNOLOGIA .NET

DELEON GALVIN

BLUMENAU
2004

2004/2-11

DELEON GALVIN

PROTÓTIPO DE SISTEMA CRM PARA DISPOSITIVOS MÓVEIS UTILIZANDO A TECNOLOGIA .NET

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

Prof. Francisco Adell Péricas

BLUMENAU
2004

2004/2-11

PROTÓTIPO DE SISTEMA CRM PARA DISPOSITIVOS MÓVEIS UTILIZANDO A TECNOLOGIA .NET

Por

DELEON GALVIN

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

Presidente:	Prof. Francisco Adell Péricas, Msc - Orientador, FURB
Membro:	Prof. Marcel Hugo, FURB
Membro:	Prof. Wilson Pedro Carli, FURB

Blumenau, 17 de novembro de 2004

Dedico este trabalho a todos que me acompanharam nesta longa jornada, principalmente ao meus pais, e a todos que acreditaram nas minhas palavras, quando falei que um dia iria chegar ao fim vitorioso.

O analfabeto do século XXI não será aquele que não pode ler nem escrever, mas sim aquele que não puder aprender, desaprender e reaprender.

Alvin Tofler.

AGRADECIMENTOS

Aos meus pais, que sempre estiveram dando força e apoio nos momentos mais difíceis, onde no horizonte apenas desistir era a única visão alcançada.

Meus irmãos que sempre estiveram presentes e auxiliando nos momentos mais necessários.

Aos meus amigos e colegas, nas ajudas e apoios necessários, onde sempre estiveram prontos para dar uma força e pelas festas e boas lembranças proporcionadas durante este período em que estivemos juntos na universidade.

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

RESUMO

Este trabalho apresenta a especificação e o desenvolvimento de um protótipo de software para integração e troca de dados com um aplicativo cliente/servidor de uma empresa através de dispositivos móveis. A aplicação visa auxiliar o trabalho de relacionamento entre empresa e clientes diretamente no campo, disponibilizando informações importantes para o usuário sobre seus clientes e também fazendo a parte de força de vendas no *front*. Este sistema demonstra a utilização das novas tendências de tecnologia a fim de auxiliar os negócios de uma empresa e facilitar e agilizar o poder de negociação e atendimento a clientes.

Palavras chaves: CRM; Dispositivos móveis; .NET; Web Service.

ABSTRACT

This project presents a software prototype's specification and development for the integration and data exchange with a client/server application for a company through mobile devices. The application aims to assist directly the efforts of creating relationship between company and customers in the field, supplying important information for the user about its customers and helping the sales force as well. This system shows the use of the new trends of technology in order to assist the company businesses and upgrade the power of negotiation and customers service

Key-words: CRM; Mobile devices; .NET; Web Service.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

.NET – dot NET

API – *Application Programming Interface*

B2B – *Business to Business*

B2C – *Business to Consumer*

BCC – Curso de Ciências da Computação – Bacharelado

CE – *Compact Edition*

CIC – Centro de Iteração com Cliente

CLI – *Common Language Interface*

CLR – *Common Language Runtime*

CLS – *Common Language Specification*

CRM – *Customer Relationship Management*

CTS – *Common Type System*

DLL – *Dynamic Link Library*

DSC – Departamento de Sistemas e Computação

ECMA – *European Computer Manufacturers Association*

EDI – *Exchange Data Integration*

ERP – *Enterprise Resource Planning*

EXE – Executável

GB - Gigabyte

JIT – *Just in Time*

MRP - *Material Requirement Planning*

MSDE – *Microsoft Database Engine*

MSIL – *Microsoft Intermediate Language*

PC – *Personal Computer*

PDA – *Personal Digital Assistant*

RAM – *Random Access Memory*

SMP – *Symmetrical Multiprocessing*

SOAP – *Simple Object Access Protocol*

SQL - *Structure Query Language*

TI – Tecnologia de Informação

VB – *Visual Basic*

XML – *Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

O dinamismo e a capacidade de mutação do mundo dos negócios está cada dia mais exigindo que pessoas consigam efetuar mais tarefas em menor tempo, visando buscar a perfeição e a execução de atividades com excelência e qualidade.

Hoje a disputa entre as empresas não se trava mais na qualidade dos produtos disponibilizados, pois este já é um fato consumado, onde produtos sem qualidade não possuem mais espaço no mercado, mas sim na capacidade de uma corporação em atender e encantar seu cliente de tal forma que consiga criar vínculos e um laço afetivo fidelizando este cliente e conquistando sua confiança em sua marca.

Mas para isto, não basta apenas ser um bom vendedor, mas sim conhecer seu cliente a fundo, buscando nas informações a arma certa na hora de encantá-lo e atendê-lo da melhor forma possível. Para auxiliar neste processo, dispõe-se de algumas teorias que poderão com certeza ser o diferencial neste momento.

O *Customer Relationship Management* (CRM) ou Gerenciamento de Relacionamento com o Cliente pode ser descrito da seguinte forma:

> Uma estratégia de negócios para selecionar e administrar os clientes, buscando otimizar o valor em longo prazo. CRM requer uma filosofia e cultura empresarial centrada no cliente para dar suporte aos processos eficazes de *marketing*, vendas e serviços. Os aplicativos de CRM podem ajudar na eficaz gestão de relações com os clientes, desde que a organização possua a liderança e a cultura corretas (GREENBERG, 2001, p. 62).

Quanto mais próximos do cliente puder-se estar, utilizando e gerando estas informações para conhecer e analisar as melhores perspectivas, maior será o resultado dos trabalhos proferidos. Portanto, precisa-se aproximar mais os aplicativos de apoio às pessoas que tem maior contato com nossos clientes.

Porém as poucas soluções que atendem atualmente o mercado neste âmbito pecam pela falta de possibilidades de configuração das reais necessidades do usuário dos sistemas, pois quem trabalha em campo, mantendo contato com clientes o tempo todo, precisa de mobilidade, praticidade e flexibilidade.

Os sistemas de CRM da atualidade possuem fortes características cliente/servidor sem pensar na necessidade de poder ser implementado em dispositivos móveis, tendo também pouca integração com outros produtos e na maioria das vezes não apresentando soluções móveis menores que um *notebook*, ou até mesmo nem possuindo uma solução que seja comportada por dispositivos móveis como *Pockets PC's*.

Para minimizar estes problemas, pode-se lançar mão das tecnologias inovadoras que estão surgindo como fortes candidatos a soluções para o mundo de *e-Business* e CRM, hoje nomeados genericamente como dispositivos móveis, juntamente com as últimas palavras em tecnologia de desenvolvimento e concepção de aplicações neste ambiente.

Entre as tecnologias do momento estão os dispositivos móveis, hoje com características cada vez mais próximas de um computador pessoal normal, tratando-se de processamento de dados, mas possuindo dimensões que possibilitam seu transporte até mesmo no bolso. Na parte de trocas de informações, a partir do momento que estamos conectados a uma rede, podemos facilmente através de *Web Services* estar efetuando a troca de dados entre o representante de campo e a base central de uma empresa, fornecendo sempre informações atualizadas e precisas no mais curto espaço de tempo facilitando o trabalho e a tomada de decisões tanto do representante quanto da empresa.

Verificando a precariedade de estudos direcionados a esta área apresenta-se neste trabalho uma análise e desenvolvimento de um protótipo de *software* de CRM focando o segmento industrial metalúrgico, rodando em dispositivos *Pocket PC* com suporte a atualização de base de dados via procedimento remoto consumindo um *Web Service* disponibilizado especialmente para esta função, utilizando as inovações de desenvolvimento disponibilizadas pela plataforma tecnológica .NET com a linguagem *Visual Basic.NET*, sistema operacional *Windows CE* e banco de dados *SQL Server 2000 CE*. O *software* deverá prover suporte à aplicação neste contexto, configurando de tal forma que possa atender a necessidade de mobilidade, praticidade e agilidade que estes usuários necessitam para conseguirem o máximo desempenho, com o menor custo.

1.1 OBJETIVOS DO TRABALHO

O objetivo deste trabalho é desenvolver um protótipo de um sistema CRM para dispositivo móvel para atender o segmento industrial metalúrgico, com base de dados local *stand alone*, atualizada via *Web Service* por serviço de conexão a rede via internet disponível para o equipamento.

Os objetivos específicos do trabalho são:

a) facilitar o trabalho do responsável pela representação da empresa em campo, disponibilizando de maneira prática as informações referentes a clientes;
b) aumentar a velocidade nas transações entre cliente e empresa;
c) agregar valor ao produto, simbolizado pela qualidade do atendimento prestado ao cliente;
d) compartilhar informações em um curto espaço de tempo melhorando as tomadas de decisões estratégicas da empresa;
e) fornecer mobilidade e flexibilidade na utilização de ferramentas de CRM.

.

1.2 ESTRUTURA DO TRABALHO

A estrutura deste trabalho está apresentada em capítulos, divididos em cinco principais temas.

O primeiro capítulo apresenta uma visão geral sobre os assuntos a serem tratados e a importância do CRM para os negócios empresariais.

Mais adiante no segundo capítulo detalha-se a estratégia de *Customer Relationship Management* (CRM) e qual sua importância para a corporação que se dispõe a trabalhar e implantá-la.

A tecnologia utilizada e a importância e diferencial são apresentados no terceiro capítulo, onde são detalhados os pontos importantes que levam a utilização da plataforma .NET como solução para a necessidade apresentada

No quarto capítulo é apresentada a solução de hardware para suportar o sistema onde são detalhadas as informações referentes aos dispositivos moveis disponíveis no mercado atualmente.

O quinto capítulo descreve todo o desenvolvimento do trabalho, incluindo a especificação, ferramentas utilizadas e metodologia de implementação.

E finalizando apresenta-se no capítulo seis as conclusões que se chegou com o desenvolvimento deste trabalho e são descritas as finalizações.

2 CONCEITO DE CRM

CRM é uma tecnologia que direciona e organiza o gerenciamento do relacionamento com os clientes. Este gerenciamento visa obter o máximo de informações do cliente, permitindo a manipulação destes dados, com o objetivo de ter-se um tratamento exclusivo e personalizado a cada um deles.

Conforme a afirmação de Ronni T. Marshak, Vice Presidente da Seybold Group, "CRM tornou-se a discussão do momento dentro do *e-business*. E parece haver mais definições de CRM do que fornecedores tentando vender pacotes para aproximação com clientes" (GREENBERG, 2001, p.119).

Tentando gerar um conceito básico para explicar o CRM, Robert Thompson, presidente da Front Line Solutions, Inc, colocou em seu site a seguinte definição:

> "A gestão de relacionamento com o cliente é uma estratégia de negócios para selecionar e administrar os clientes, buscando otimizar o valor em longo prazo. CRM requer uma filosofia e cultura empresarial centrada no cliente para dar suporte aos processos eficazes de *marketing*, vendas e serviços. Os aplicativos de CRM podem ajudar na eficaz gestão de relações com os clientes, desde que a organização possua a liderança, as estratégias e a cultura corretas."(GREENBERG, 2001, p. 62).

Para a área de TI, CRM é um *software* que integra os módulos de automação de força de vendas, gerência de vendas, *telemarketing* e *call centers*, serviço de suporte e atendimento eletrônico, possibilitando traçar estratégias de negócios voltadas ao encantamento do cliente e à supremacia da corporação para com o cliente antecipando necessidades e identificando oportunidades junto ao cliente.

O estabelecimento e o gerenciamento das relações com os clientes são, antes de tudo, uma postura estratégica, não uma categoria tecnológica. Da mesma forma que a empresa estabelece suas metas, estratégias, planos e objetivos, ela deve também determinar como as relações com seus clientes serão tratadas passo a passo.

Sendo assim, o CRM deve começar como uma estratégia dos negócios, desencadeando uma mudança organizacional e nos processos de trabalho, as quais mais à frente serão suportadas pela tecnologia da informação.

Dessa forma, garante-se o sucesso do processo de implantação, pois a maioria dos projetos que primeiramente focam a tecnologia, ao invés dos objetivos dos negócios, acabam fracassados, pois uma empresa nunca conseguirá chegar a uma nova estratégia de negócios a partir da automatização de seus processos.

Quando se tem a tecnologia entrando em ação no processo de CRM, precisa-se buscar o que há de mais avançado na tecnologia digital para atender bem o cliente, estando cada vez mais próximo dele com as informações geradas, e ao mesmo tempo alimentando a base de conhecimento do cliente na empresa para um processo de tomada de decisão rápido e preciso.

E a mobilidade é a palavra do momento, pois a miniaturização dos equipamentos já é uma verdade no mundo dos negócios, garantindo uma comunicação efetiva e continua entre clientes, funcionários e empresa.

Hoje a principal razão pela qual se procura uma nova tecnologia é a conveniência. Essa conveniência e eficiência empresarial resultantes vão superar os custos mensais de acesso e manutenção nas empresas, fazendo com que estas busquem cada vez mais este tipo de solução, gerando o estímulo para a criação de aplicativos para uso em dispositivos sem fio, que possam fornecer informações completas e amigáveis para vendedores, clientes, e os gerentes.

Hoje se pode dividir a questão de *software* de CRM em três campos distintos, sendo eles o CRM operacional, o CRM analítico e o CRM colaborativo, onde cada um deles possui sua parcela de contribuição para a conquista do resultado final.

2.1 CRM OPERACIONAL

Este segmento se assemelha ás funções de um *Enterprise Resource Planning* (ERP) dentro de uma empresa.

Funções empresariais típicas envolvendo os serviços ao consumidor, gerenciamento de pedidos, faturamento, contabilidade, cadastros e contatos e a automação de força de vendas ficam classificadas como atividades deste seguimento. Pode ser considerado como um dos primeiros passos do CRM.

É neste momento em que se está armazenando as informações do cliente, pois todas as interações com o cliente ficam englobadas nesta divisão do CRM, sendo de suma importância o correto gerenciamento desta etapa do processo, para se obter informações precisas e produtivas.

E neste momento, precisa-se aproximar ao máximo a tecnologia dos usuários, buscando repassar as informações do cliente no menor espaço de tempo para a base central, permitindo que os usuários tanto na ponta produtiva quanto na ponta estratégica possuam total confiança nas informações e que tomem suas decisões sobre os seus clientes de forma confiante. As informações coletadas devem ser repassadas de forma que possam ser recebidas e processadas no menor espaço de tempo. Um exemplo de processo bem sucedido por exemplo, seria a automação de força de vendas, possibilitando envio de pedidos e dados de cliente para a empresa de forma rápida e segura com toda mobilidade e confiança, e ao mesmo tempo garantir que este usuário na ponta receba os resultados deste pedido enviado para uma ação pró ativa com o cliente.

Este processamento das informações se dá pela possibilidade e necessidade de integração do sistema de CRM com outros aplicativos encarregados da parte financeira e recursos humanos, como por exemplo, o ERP.

Depois de toda esta integração, pode-se ter garantido que o sistema estará efetuando desde o gerenciamento dos contatos ao rastreamento de todos os pedidos do cliente, possibilitando o conhecimento de todas as ações do nosso cliente, e também podendo passar para ele todas as posições dele referente à empresa.

Porém neste mesmo passo tem-se uma afirmação não muito animadora, pois segundo Greenberg (2001), o índice de insucesso de implantações de CRM variam de 55% a 65% e este insucesso já é identificado na implantação e integração do CRM com os aplicativos e plataformas.

2.2 CRM ANALÍTICO

CRM analítico diz respeito à captação, armazenagem, acesso, pesquisa e interpretação das informações de clientes para os usuários do sistema (GREENBERG, 2001, p.67).

No CRM analítico, todos os dados coletados referente ao cliente, desde histórico financeiro, até mesmo as datas de aniversário dos contatos no cliente, são armazenados em um banco de dados de clientes e então a partir de ferramentas especificas com a finalidade de mineração de dados, são utilizadas centenas de algoritmos para analisar e interpretar todas as informações geradas conforme a necessidade de informação do usuário.

É a fonte de inteligência de toda a estratégia CRM, pois, por exemplo, pode ser feito o reconhecimento dos mais valorosos clientes e assim balancear os esforços e atenções aos clientes corretos, melhorando relacionamentos e efetuando promoções sobre este reconhecimento. Nesta fase é que são geradas as necessidades, e as áreas envolvidas no relacionamento direto com o cliente podem tomar ações diretas sobre as informações coletadas dos clientes.

Os sistemas são alimentados com informações relativas a estes clientes, desde os sistemas cliente/servidor ou até mesmo os computadores móveis utilizados pelos vendedores em campo. Porém estas informações podem acabar se pulverizando em muitos sistemas, pois se pode ter vários sistemas diferentes dentro de uma empresa, como por exemplo, sistemas de contabilidade, sistema de entrada de pedidos, sistema MRP ou ERP, sistema de gerenciamento de estoques e ativos, sistemas de distribuição, sistemas de Relações Humanas, sistemas de gerenciamento de interação com clientes onde todos podem ter forma de utilização, aplicação e estruturas de dados operacionais diferentes e incompatíveis entre si.

Segundo Swift (2001), para contornar este problema, atualmente empresas líderes têm utilizado *data warehouses*, com foco no cliente e que coloca à parte os seus tradicionais bancos de dados operacionais.

O *data warehouse* transformou-se na base sobre a qual objetivos específicos de negoócio podem ser alcançados: são muito mais que apenas um simples depósito para armazenar dados (SWIFT, 2001, p.78).

2.3 CRM COLABORATIVO

Mais adiante, tem-se o CRM colaborativo, área esta que compreende um centro de comunicação pessoal entre a empresa e o cliente, representado pelos *call centers*, sistemas de

correio de voz, ou ainda os sistemas web de interação direta com o cliente. Em cima dessas teorias, muitas funções podem ser adotadas para encantamento do cliente via este canal.

Segundo Greenberg (2001, p. 70), na medida que a população sente-se cada vez mais confortável utilizando a internet de maneira segura, é provável que a quantidade de negócios que costumava fazer por telefone ou mesmo pessoalmente passe a ser feita via internet.

Conforme apresentado por Greenberg (2001, p.73), um estudo realizado pela Roper/AOL Cyberstudy no início de 2000, sobre hábitos de compras *online*, comprovaram que em 1998, 31% dos navegadores da rede compravam quase regularmente alguma coisa *online*, e que no ano seguinte esse valor subiu para 41%. Segundo o estudo ainda, esse aumento se deveu ao aumento de mulheres navegando na rede. No mesmo estudo, foi verificado que na internet as mulheres representavam 24% dos compradores *online* e em 1999 subiram para 37% do total. Sem contar que estudos em meados de 2000 comprovaram que 52% dos internautas já são mulheres.

Sendo assim a migração para este ambiente Web do CRM significa crescimento e velocidade apoiando e interagindo diretamente com o cliente. Outra grande idéia são os *call centers*, que são sem dúvida ainda o melhor meio de interação direta com o cliente.

> Muitos estudiosos chegam a dizer que na aparição dos *call centers* é que realmente começou a era do CRM, apesar de podermos também dizer que a automação da força de vendas seja o verdadeiro avô mais provável do CRM. Mas podemos dizer que o segmento de serviços a clientes é um tio-avô do CRM. (GREENBERG, 2001, p. 214).

Chamar um centro de interações com cliente de CRM não é um truque de *marketing*. Sabe-se que em breve o cliente não poderá mais ser considerado como um transmissor ou receptor de informações. A atividade colaborativa é um dos aspectos que fazem disto uma aplicação de CRM. O cliente interage diretamente com a empresa por meio de um representante de serviços e uma variedade de canais de comunicação, e todos utilizam ferramentas que tornam essas interações valiosas. O cliente pode interagir com o *web site*, por meio de aplicativos de auto-serviço ou outros.

Porém um dos problemas nesse mundo de expectativas em rápidas mudanças por parte dos clientes é que a evolução de um *call center* para um centro de iteração com clientes ou um

centro de contatos é lenta embora a sua taxa de mudanças seja acelerada. Segundo Greenberg (2001, p. 215) a Gartner Group estima que apenas 20% dos *call centers* existentes tenham incorporado um ponto de contato ao vivo via internet, ou algum tipo de resposta automática por e-mail, por volta de 2002. A previsão para 2005 é que este número seja de 70%, sendo que em 2004 esses números ainda não superam 50%. Sendo assim significa que a maior parte dos *call centers* ainda precisarão de operadores humanos com toda certeza.

2.4 CONHECENDO, SEGMENTANDO E MANTENDO CLIENTES

No passado era mais fácil buscar os clientes na concorrência provando apenas ter um produto melhor que o dela para oferecer. Porem análises recentes provam que esta forma de tratar clientes trazem prejuízos, pois as empresas gastam em média cinco vezes mais na aquisição de um novo cliente para sua carteira do que o valor gasto para reter e manter um cliente já existente.

Estudos feitos comprovam que quando um relacionamento entre um cliente e companhia aumenta, os lucros sobem. As empresas podem aumentar os lucros em 100% retendo somente 5% de seus potenciais clientes (SWIFT, 2001, p.76).

Sendo assim a capacidade de fazer perguntas certas que se relacionem com uma estratégia bem fundamentada e pensada é o sucesso para o *marketing* da empresa.

As perguntas feitas são fundamentais, passíveis de serem respondidas com dados disponíveis, porém descobrir quais os dados e obter as respostas exatamente como deseja não são tarefas fáceis.

Portanto, para fazer a oferta certa para o cliente certo, necessita-se conhecer quem são os clientes.

A empresa focada diretamente no cliente, com todas as informações armazenadas corretamente e de forma concreta e robusta, com certeza trará resultados totalmente positivos, atingindo os clientes certos com as promoções certas, no momento certo e pelo canal certo.

Trabalhando desta forma conquista-se uma redução de custos, por passar a trabalhar com marketing direto e focado no cliente, mostrando conhecer exatamente qual a necessidade do cliente.

2.5 CRM E TECNOLOGIA DA INFORMAÇÃO

O conhecimento detalhado dos clientes e não somente dos dados brutos relacionados com transações e pagamentos financeiros, é o que as empresas bem-sucedidas utilizam para reter clientes lucrativos.

A transformação de dados brutos em informação que podem ser consultadas é obrigatória para a criação de um ambiente para a tomada de decisões de negócios compartilhada e inovadora.

> As principais funções como *marketing*, comunicações com os clientes e serviços a clientes, planejamento de vendas, desenvolvimento de produto, gerenciamento de distribuição, análise financeira e custos, avaliação de riscos e gerenciamento de canal precisam estar ligados por recursos de informação e processos analíticos para se ter uma visão precisa, oportuna e completa do cliente (SWIFT, 2001, p. 68).

O poder de conhecimento do relacionamento com o cliente leva a maiores níveis de respostas e lucratividade ao negócio. Os recursos despendidos e as oportunidades para gerenciar o cliente e os canais aumentarão se os sistemas de informação e as capacidades de apoio à decisão forem orientados para uma visão multidepartamental ou de negócios múltiplos. Isso significa uma abordagem dirigida a toda empresa, para extrair e transformar sistemas de informação em uma info-estrutura voltada para o cliente.

A função das organizações de tecnologia da informação (TI) é fornecer estratégia, infraestrutura, tecnologia, informações e processos para facilitar o trabalho da empresa na coleta, gerenciamento e proteção das informações. Inclusive outra função também é conhecer todas as fontes possíveis para obtenção de informações e a orientação da melhor forma de utilização das melhores formas de tecnologia de informação.

Segundo Bretzke (2000, p.143), independente do tipo de técnica utilizada, é preciso que a empresa assegure que cada atividade relacionada com o contato com o cliente seja parametrizada e contemplada nos *Data Warehouses* de *Marketing*, e que todo o

acompanhamento seja devidamente mapeado em todo o ciclo de relacionamento com o cliente.

Sendo assim, Swift (2001, p.78) comenta que as organizações de TI montaram unidades de negócios para forjar um ambiente de *Data Warehouse* que transformam informações em conhecimento e fornecem um amplo acesso a dados históricos muito detalhados.

Todo este trabalho conclui a definição de um sistema que irá capturar dados de transações de diversas fontes díspares em toda a organização e direcioná-los a um repositório único conciliando e processando todo o dado recebido separando por áreas de interesse previamente definidas.

2.6 SISTEMAS DE INFORMAÇÃO

É um conjunto de elementos interdependentes (subsistemas), logicamente associados, para que de sua interação sejam geradas informações necessárias a tomadas de decisão (CAUTELA; POLLONI, 1978, p. 17). Seu objetivo é, portanto, gerar informações para tomadas de decisões, ou seja, gerir e transformar massa bruta de dados em informações que possuam um perfil analítico e transparente da situação.

A informação gerada deve ser de alta qualidade, filtrada em níveis de decisão, por exemplo, de acordo com os níveis hierárquicos empresariais que irão utilizá-la, apresentando desde informações de mais baixo nível para tomadas simples de decisão em um grupo de trabalho como até mesmo informações condensadas preparadas para uma tomada de decisão estratégica para o alto escalão da empresa.

Sendo assim, segundo Cautella; Polloni (1978, p.18) pode-se classificar uma informação como ideal quando atende os seguintes requisitos:

a) clareza – apresentar o fato com clareza, não o mascarando entre fatos acessórios;
b) precisa – a informação deve ser de um alto padrão de precisão e nunca apresentar termos como: “por volta de...”, “cerca de...”, “mais ou menos...”;
c) rápida – chegar a ponto de decisão em tempo hábil para que surta efeito na referida decisão. Uma informação pode ser muito clara e precisa, mas, se chegar atrasada ao momento de decisão, já perdeu a razão de ser;

d) dirigida – a quem tenha necessidade dela e que irá decidir com base nessa informação.

Os sistemas de informação essencialmente transformam a informação em uma forma utilizável para a coordenação de trabalhos de uma empresa ajudando empregados ou gerentes a tomar decisões, analisar e visualizar assuntos complexos e resolver outros tipos de problemas (LAUDON; LAUDON, 1999, p. 4).

Os sistemas de informação possuem um ciclo de três atividades básicas: entrada, processamento e saída. A entrada envolve a captação ou coleta de fontes de dados brutos de dentro da organização ou de seu ambiente externo. O processamento envolve a conversão desta entrada de dados bruta em uma forma mais útil e apropriada. A saída, envolve a transferência da informação processada às pessoas ou atividades que a utilizarão. Os sistemas de informação são responsáveis também pela armazenagem da informação nas mais variadas formas até o momento que ela seja necessária à utilização. Existe ainda a realimentação que é a saída que retorna aos membros adequados da organização a resposta obtida com a informação utilizada, ou seja, é o *feedback* referente utilização das informações geradas.

Um sistema de informação é uma parte integrante de uma organização e é um produto de três componentes: tecnologia, organizações e pessoas. (LAUDON; LAUDON, 1999, p.5).

Organizações moldam os sistemas de informações de várias formas óbvias. Empresas são organizações formais. Consistem em unidades especializadas com divisão nítida de mão de obra e especialistas nas mais diversas funções. Possui uma disposição hierárquica, onde existem os gerenciáveis e os gerenciados. Procedimentos em todos os níveis são padronizados e também formais, todos em torno de uma causa única de sucesso final. Neste caso os sistemas de informação analisam e fornecem informações dos mais variados tipos para poder atender toda a gama de informação necessária a todos os interessados na empresa.

As pessoas usam as informações de sistemas baseados em computadores em seus trabalhos, integrando informação e trabalho em um bloco único de busca de soluções. A interação entre o homem e a máquina deve ser a mais ergonômica possível para que se tire o maior proveito das informações geradas para o auxílio na realização dos trabalhos.

A tecnologia é o meio pelo qual os dados são transformados e organizados para o uso por pessoas. Um sistema de informação pode ser desde um sistema manual simplesmente

focado em papel e arquivos até mesmo computadores e uma infraestrutura completa de armazenamento e manutenção de dados em um meio de tecnologia de informação.

A composição dos meios tecnológicos de sistemas de informação para atender todo o ciclo de entrada, processamento e saída é enumerada por Laudon e Laudon (1999, p.7) da seguinte forma :

a) hardware – é o equipamento físico usado para a tarefa de entrada, processamento e saída em um sistema de informação. Consiste na unidade de processamento do computador e pelos periféricos e meios físicos que interagem estes dispositivos;
b) software – consiste em instruções pré-programadas que coordenam o trabalho dos componentes de hardware para que executem os processos exigidos pelo sistema de informação;
c) tecnologia de armazenamento – para armazenar e organizar os dados utilizados por uma empresa, um dos determinantes é a disponibilidade e utilidade dos dados. Ela inclui os meios físicos de armazenamento como discos rígidos ou magnéticos ou óticos, ou ainda fitas, assim como os softwares que regem a organização dos dados nesse meio físico;
d) tecnologia de comunicação – é utilizada para conectar partes diferentes de hardware e para transferir dados entre pontos via redes. Uma rede liga dois ou mais computadores entre si para transmissão de dados, voz, imagens, sons, vídeos ou para compartilhar os mais variados recursos de informática.

Nenhum sistema sozinho rege todas as atividades de uma empresa inteira. As empresas têm diferentes tipos de sistemas de informação para enfocar diferentes níveis de problemas e diferentes funções dentro de uma organização (LOUDON;LOUDON, 1999, p. 27).

> Cada área funcional de uma firma desenvolve sistemas: há sistemas de fabricação e produção, sistemas de finanças e contabilidade, sistemas de vendas e marketing e sistemas de recursos humanos. Os sistemas também servem diferentes níveis: sistemas em nível estratégico ajudam os planos da gerencia sênior, sistemas para as gerencias intermediarias ajudam no controle das atividades diárias das organizações; sistemas de conhecimento ajudam engenheiros e funcionários de escritórios e sistemas operacionais são usados em expedição e produção. (LOUDON;LOUDON, 1999, p. 27)

Nos dias atuais um dos principais focos de um sistema de informação é galgar vantagem competitiva frente os concorrentes.

Esses sistemas são considerados sistemas de informações estratégicas, porque se encontram em resolver problemas relacionados à prosperidade da empresa em longo prazo e sua sobrevivência. Tais problemas podem significar a criação de novos produtos e serviços, o estabelecimento de novas relações com clientes e fornecedores, ou a descoberta de meios mais eficientes e mais eficazes de se administrar as atividades internas da empresa.

> O objetivo desses sistemas é fornecer soluções que permitirão as empresas derrotar e frustrar seus concorrentes. Sistemas como este tem impacto significante na empresa com uma ação de grande alcance e profundamente arraigados; eles mudam fundamentalmente os objetivos, produtos ou relações internas e externas da empresa. (LOUDON; LOUDON, 1999, p. 27).

3 MICROSOFT.NET

Com o seu lançamento em junho de 2000, a Microsoft anunciava a sua iniciativa .NET, dando uma nova visão e ampliando horizontes incluindo a internet e a WEB no desenvolvimento, engenharia e uso de *softwares*. Um dos aspectos importantes da estratégia .NET foi a independência de plataforma ou linguagem específica, onde os desenvolvedores podem utilizar qualquer linguagem compatível .NET para desenvolvimento, não se prendendo mais a apenas uma linguagem.

Outro passo importante foi a tecnologia ASP.NET a qual permite desenvolvedores criarem aplicativos para a WEB.

A Microsoft .NET expande os conceitos de Internet e sistema operacional, transformando a própria Internet na base de um novo sistema operacional.

Em última instância, isso permitirá aos desenvolvedores criarem programas capazes de transcender as limitações dos dispositivos e atrelar completamente seus aplicativos à conectividade da Web.

A .NET deixou de depender basicamente do *hardware*, possibilitando ir além das fronteiras da infraestrutura conhecida habitualmente. Seus dados passam a residir na Internet e não mais em apenas uma máquina, podendo ser acessados de qualquer PC, *laptop*, telefone celular ou PDA e serem integrados entre diversos aplicativos.

Um conceito fortemente comentado na estratégia .NET é o fato dela estender a reutilização de *software* à internet, permitindo que os programadores concentrem–se em suas especialidades sem ter de implementar todo o componente de todo aplicativo. Ao invés disto as empresas podem comprar *Web Services* e dedicar tempo e energia para desenvolver seus próprios produtos.

Da mesma forma que a programação visual tornou-se popular pelo fato de utilizar componentes pré-empacotados como botões, caixas de diálogos e barras de rolagem, os programadores utilizando *web services* podem buscar bancos de dados, segurança, autenticação, armazenamento de dados, tradução de linguagens e outros, sem saber os detalhes de implementação destes componentes.

Um bom exemplo que se pode apresentar é uma única aplicação que poderia gerenciar pagamento de contas, restituição de impostos, empréstimos e investimentos, utilizando *web services* de várias empresas. Um comerciante *online* poderia utilizar os *web services* para pagamentos com cartão de crédito *on-line*, autenticação de usuário, segurança da rede e banco de dados de inventário para criar um site *Web* de *E-Commerce* (DEITEL, 2004, p. 13).

As chaves para estas integrações são o *eXtensible Markup Language* (XML) e o *Simple Object Access Protocol* (SOAP) que possibilita que os *Web Services* se comuniquem.

Outro forte conceito da .NET é a questão do acesso universal aos dados, trabalhando fortemente a idéia de sincronização, onde dois arquivos iguais em dois locais remotos, caso um deles esteja defasado em informações deve ser imediatamente atualizado, mantendo sempre os mesmos dados nos dois lados. Na realidade, a forma mais correta neste caso seria ter os dados centralizados em um único ponto, onde todos os outros consumidores destas informações poderiam estar buscando via *Web Service*.

3.1 WEB SERVICES

Os *Web Services* ou Serviços da Web (algumas vezes chamados de XML *Web Services*) foram criados para promover a reutilização de *software* em sistemas distribuídos. Sistemas Distribuídos permitem que aplicativos sejam executados ao longo de múltiplos computadores em uma rede (DEITEL, 2003, p. 341).

Ainda pode-se encontrar uma definição parecida como sistema que pode ser descrito como qualquer funcionalidade acessível na Internet, geralmente utilizando troca de mensagens com a eX*tensible Markup Language* (XML) utilizando os protocolos de comunicação existentes (SCRIBNER, 2002, p. 144).

Basicamente trata-se de uma classe armazenada em uma máquina, que por sua vez pode ser acessada em outra máquina em uma rede. Esta relação dá à máquina servidora o nome de máquina remota (DEITEL, 2003, p. 357).

Os principais e mais vantajosos usos da *Web Service* são as integrações *Business to Business* (B2B) onde este paradigma inclui a operação entre várias empresas e plataformas. Por mais de duas décadas, desenvolvedores buscam integrar processos de negócios utilizando

a combinação entre sistemas e *softwares* e protocolos de comunicação (SCRIBNER, 2002, p. 170).

No início, o *Exchange Data Integration* (EDI) foi o mais famoso de todos, perdurando por anos. Apesar de útil e de ter atendido esta área a contento, sua implantação despendia custos que somente eram possíveis de serem bancados por grandes empresas.

A XML atrelada aos *Web Services* vem com o conceito de mudar este cenário, padronizando e facilitando os custos e implantação, por fornecer a padronização que não existia na EDI, baixando custos e tempo de implantação.

3.1.1 Arquitetura Conceitual de um Web Service

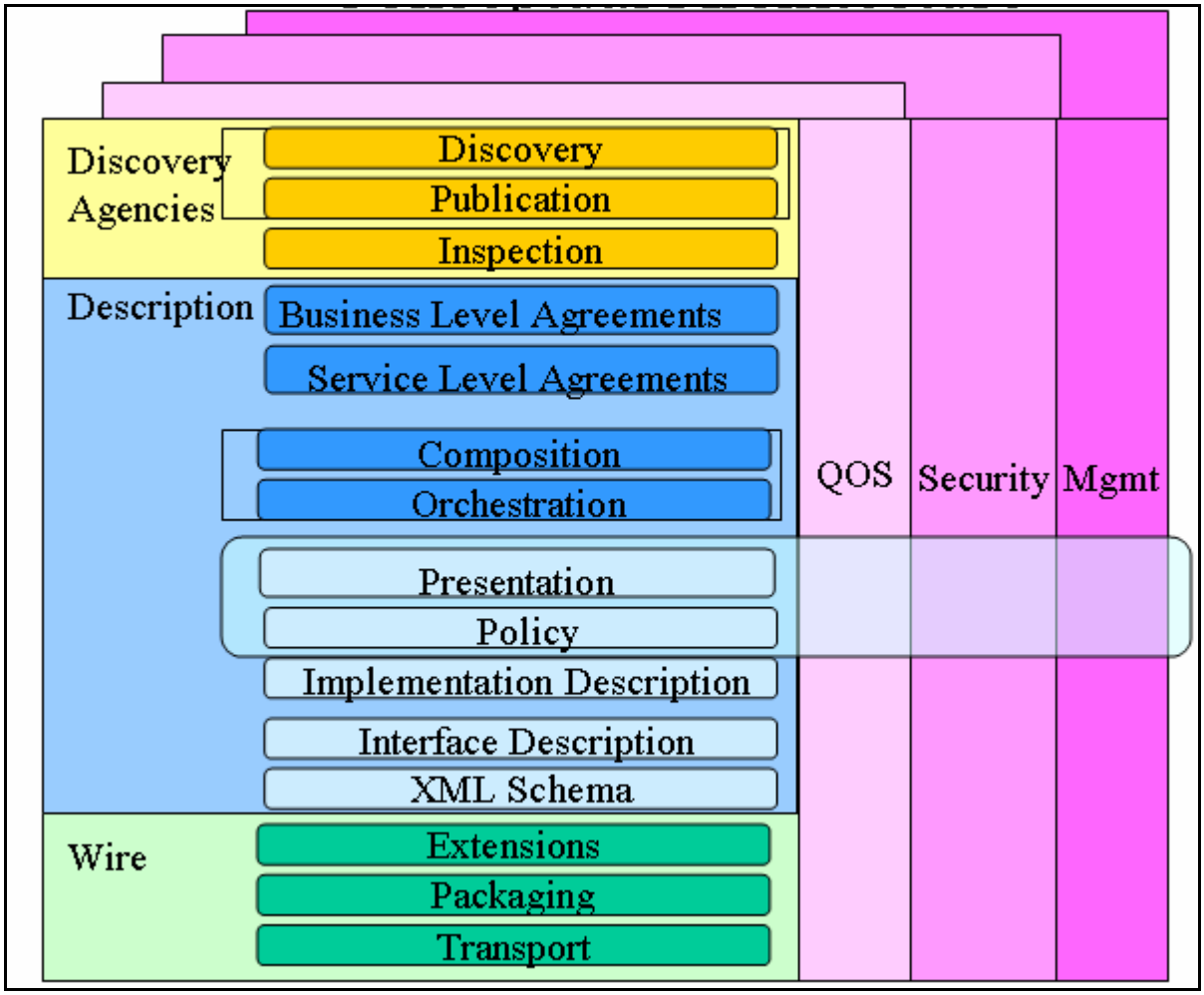
A arquitetura padrão de um *Web Service* pode ser descrita em diversas camadas. Para as diferentes operações de descoberta, invocação e publicação de serviços, são necessárias as diversas camadas, as quais compõem a chamada Pilha de *Web Service*.

Para um *Web Service* ser interessante, é necessário que este esteja publicado em uma rede, e compartilhado aos usuários afins, utilizando um protocolo comum a todos sendo a base da camada de pilha do Web Service, como por exemplo, o *Hypertext Transfer Protocol (*HTTP), mas também podendo ser utilizado o *Simple Mail Transfer Protocol* (SMTP) ou o *File Transfer Protocol* (FTP).

A troca das mensagens é baseada em XML, e para esta segunda camada, o protocolo SOAP é o selecionado por possuir a maior portabilidade tendo seu conceito de envelope de mensagens centralizado em documentos ou chamadas remotas de procedimento (RPC).

Na descrição de serviços, é utilizada a WSDL. Este padrão se faz necessário para a interoperabilidade entre os *web services*. É ela quem define a interface de comunicação com o serviço assim como a dinâmica de interações com ele.

A figura 1 mostra graficamente a estrutura em camadas necessária para o funcionamento de um *web service.*



Fonte: Champion, Ferris, Newcomer, Orchard (2002)

Figura 1 - Pilha de protocolos *Web Service*

3.1.2 XML

A linguagem *eXtensible Markup Language* (XML) foi desenvolvida pelo *World Wide Web Consortiun* (W3C) com um perfil de *Standard General Markup Language* (SGML) desenvolvido especialmente para aplicações WEB (SILVA, 2004, p. 15).

A partir desta linguagem é possível definir extensões de linguagens de marcação, através do uso de novas *tags*, onde seus dados são representados dentro de uma forma de fontes semi-estruturados aproveitando-se das facilidades e da flexibilidade oferecida com as novas definições de *tags* e aninhamentos.

A principal diferença de um documento HTML e de um documento XML está na possibilidade da associação de um *Document Type Definition* (DTD), onde consultas não dizem respeito apenas a dados, mas também a consultas de novas estruturas através de linguagens específicas para consultas sobre fontes de dados semi-estruturados. (SCRIBNER, 2002, p. 77).

Outra característica marcante é a utilização de *namespaces*, que são uma forma simples e direta de qualificar nomes de elementos e atributos encontrados dentro de um XML associando através de seus prefixos a espaços de nomes associados a alguma *Universal Resourse Identifier* (URI), *Uniform Resource Locator* (URL) ou *Uniform Resourse Number* (URN).

3.1.3 SOAP

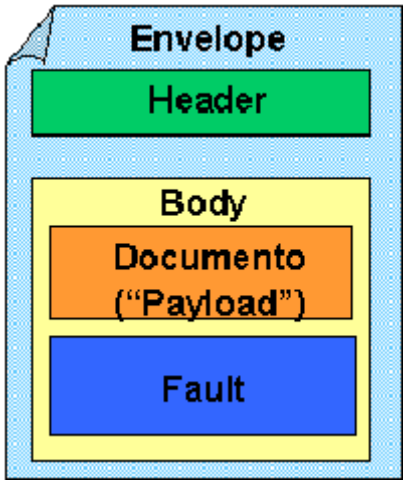
Simple Object Access Protocol (SOAP) é um protocolo projetado para invocar aplicações remotas através de uma *Remote Procedure Call* (RPC), ou troca de mensagens em um ambiente independente de plataforma ou de linguagem de programação (CUNHA, 2002).

A sua principal utilização se dá ao fato da necessidade de interoperabilidade e utilização multiplataforma, através de uma linguagem XML e um protocolo HTTP padrões.

Sua estrutura basicamente é formada por três principais componentes:

a) envelope – toda mensagem SOAP deve contê-lo pois é o elemento raiz de um XML. Ele pode conter declarações de *namespaces* e também atributos adicionais como o que define o estilo de codificação . Um *encoding style* é quem define como os dados são descritos dentro de um documento XML;
b) *header* – é um cabeçalho opcional. Ele carrega informações adicionais como por exemplo, se a mensagem deve ser processada por um determinado nó intermediário, pois ao trafegar na rede a informação passa por diversos pontos (nós), até alcançar o destino final;
c) *body* – este elemento é obrigatório e contém o *playload*, ou a informação a ser transportada ao seu destino final. O elemento *body* pode conter um elemento adicional *Fault*, usado para carregar mensagens de *status* e erros que podem ocorrer nos nós da mensagem durante seu processo.

A figura 2 mostra como fica montado um “envelope” completo:



Fonte: Cunha (2003)
Figura 2 - Estrutura de uma mensagem SOAP

3.2 PLATAFORMA .NET

Conforme Deitel (2003), pode-se considerar a plataforma .NET o coração da estratégia .NET. Ela é quem gerencia e executa os aplicativos e serviços da WEB, contendo *a Framework Class Library*, garantindo segurança e fornecendo muitos recursos de programação.

> A plataforma .NET é a estratégia da Microsoft para distribuir *softwares* como serviço. Incluindo a isto as ferramentas para criar e operar uma nova geração de serviços, a experiência do usuário .NET para possibilitar *rich clients*, serviços de *building blocks* de construção da .NET, e *softwares* de dispositivos .NET para possibilitar uma nova geração de dispositivos de internet interligados (MSDN, 2001, p. 118).

Os detalhes da plataforma .NET encontra-se na *Common Language Specification (CLS)*, a qual contém as informações sobre o armazenamento de tipos de dados, objetos, etc. A CLS foi submetido à padronização pela *European Computer Manufacturers Association (ECMA)* tornando mais fácil criar a plataforma .NET para outros ambientes. Atualmente a

plataforma .NET existe apenas para o ambiente *Windows*, mas já possui uma versão em desenvolvimento para o ambiente FreeBSD, sistema operacional do tipo UNIX de código-fonte aberto e gratuitamente distribuído.

O C*ommon Language Runtime (CLR)* é uma outra parte central da plataforma .NET, pois é ela quem executa os programas desenvolvidos na plataforma. O procedimento de compilação para instrução de máquina é efetuado em dois passos. Primeiro compila-se numa *Microsoft Intermediate Language (MSIL)* que define as instruções para a CLR. Somente depois disto então é que realmente o código MSIL será compilado para código de máquina, criando então um aplicativo único.

A razão pela adoção deste primeiro passo de compilação para uma linguagem intermediária deve-se a necessidade de portabilidade entre sistemas operacionais, a interoperabilidade entre as linguagens e os recursos de gerenciamento de execução, como o gerenciamento de memória e de segurança tornando programas desenvolvidos na .NET independentes de plataforma, ou seja, um código uma vez escrito e compilado pode ser executado em qualquer plataforma de sistema operacional.

> A interoperabilidade de linguagens oferece vários benefícios às empresas de software. Os desenvolvedores de *Visual Basic.NET* e *Visual C++.NET* podem trabalhar lado a lado no mesmo projeto sem ter que aprender uma outra linguagem de programação – todos os seus códigos são compilados em MSIL e ligados para formar um programa. (DEITEL, 2004, p. 15).

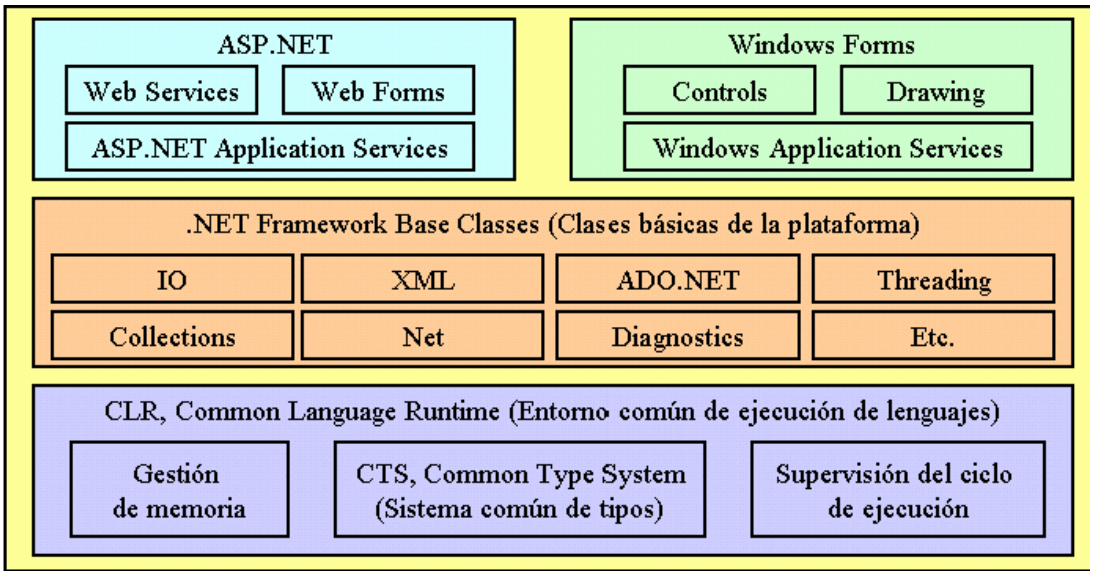
A plataforma .NET é formada por uma vasta biblioteca de classe chamada de *Framework Class Library* (FCL), que contempla uma variedade de componentes reutilizáveis poupando os trabalhos do programador e agilizando o processo de desenvolvimento.

3.3 FRAMEWORK .NET

A .NET Framework é um ambiente para a criação, instalação e execução de serviços da web, e outros aplicativos. Detalha-se nas seguintes subdivisões, sendo a primeira a *Common Language Runtime*, a qual fornece uma variedade de serviços para os aplicativos e processa a execução na plataforma .NET, independente da linguagem na qual eles foram escritos originalmente. Este serviço inclui ainda a gerência de memória e a coleta de lixo. Fornece ainda muitos serviços que ajudam na simplificação do desenvolvimento de código

melhorando a confiabilidade da aplicação e também o tratamento de exceções e de erros, unificando as depurações em um ambiente único. Este trabalho é feito de forma transparente, simplificando assim as tarefas dos programadores e administradores (ROMAN, 2002, p.218).

A figura 3 pode determinar os componentes da .NET Framework:



Fonte: Blanco (2002)

Figura 3 - Estrutura da .NET Framework

A segunda área é composta pelas *Unified Core Classes*. Esta área contém classes que fornecem facilidades que os programadores requerem para construção de uma aplicação moderna, tais como, suporte para XML, conectividade de rede e acesso a dados. Tendo estas classes unificadas, permite a um programador construir uma aplicação para o *Windows* ou para a *Web*, usando as mesmas classes. Esta consistência resulta numa melhor produtividade para o programador e uma melhor reutilização de código (MSDN, 2001, p.94).

A terceira área envolve as classes de apresentação, onde se inclui o ASP.NET para desenvolvimento de aplicações *Web*, os Serviços Web XML e os *Forms* do *Windows* para o desenvolvimento de aplicações baseadas no *Windows*, bem como de aplicações para PDA's.

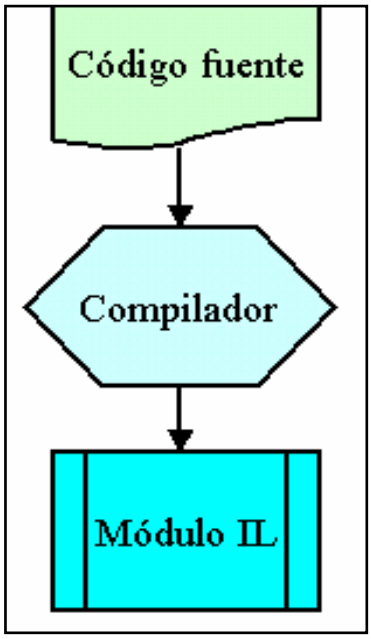
3.4 MICROSOFT INTERMEDIATE LANGUAGE

Durante o processo de compilação do código fonte, o código é compilado conforme a linguagem de programação utilizada no desenvolvimento, porém não é convertido

diretamente para código binário de execução, mas sim para um código intermediário (BLANCO, 2002, p.47).

Conforme Blanco (2002), este processo de compilação intermediária é descrito como a *Microsoft Intermediate Language* (MSIL). Esta linguagem gerada pelo compilador consiste em um conjunto de instruções independente do sistema operacional e do processador que irá executar este programa, e se ocupa diretamente com os objetos gerados no desenvolvimento, na manipulação destes objetos, nos acessos a memórias, no gerenciamento de exceções entre outros.

A figura 4 determina basicamente como é o processo de compilação da MSIL.



Fonte: Blanco (2002)
Figura 4 - Processo de compilação da MSIL

Esta compilação gera também o metadados, que transportará todas as informações da aplicação a ser executada, informações estas que serão utilizadas pela CLR para a execução final.

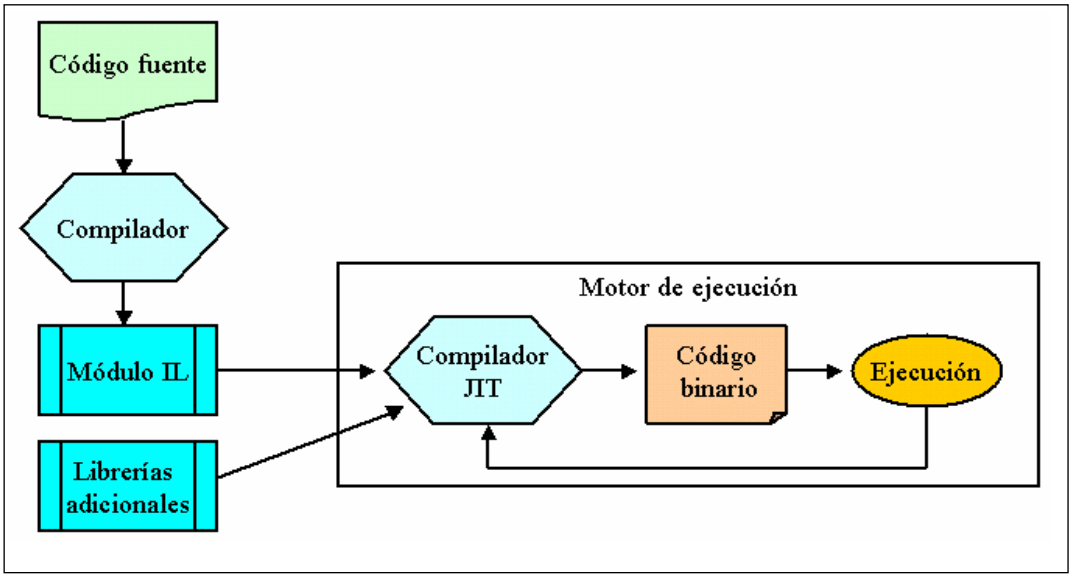
Tanto o MSIL quanto o metadados são gerados e armazenados em arquivos com extensão EXE e DLL, baseados na especificação normal da *Microsoft* para arquivos de executáveis transportáveis e arquivos de objeto comuns.

Este MSIL é totalmente independente do processador, onde apenas se preocupa em traduzir qualquer tipo aplicativo desenvolvido em uma linguagem compatível com a .NET em um código único padronizado, garantindo assim a portabilidade e a escrita de um mesmo aplicativo em diferentes linguagens. Porém em seu estado atual não é possível executá-la sendo necessário o segundo passo de compilação para código de máquina, usando um compilador *Just-in-Time* conforme se mostra a seguir.

3.5 COMPILACAO INSTANTÂNEA DA MSIL (JUST-IN-TIME)

A MSIL não pode ser diretamente executada, sendo assim, antes da execução final, esta MSIL deve ser convertida para código binário de máquina, e para isto é necessária a utilização de um compilador *Just-in-Time* (JIT *Compiler*), que esta diretamente encarregado da tradução da MSIL para código binário de máquina especifico para a arquitetura que está executando o aplicativo.

A figura 5 apresenta o funcionamento do processo de compilação de um aplicativo completo:



Fonte: Blanco (2002)

Figura 5 - Processo completo de compilação de um aplicativo .NET

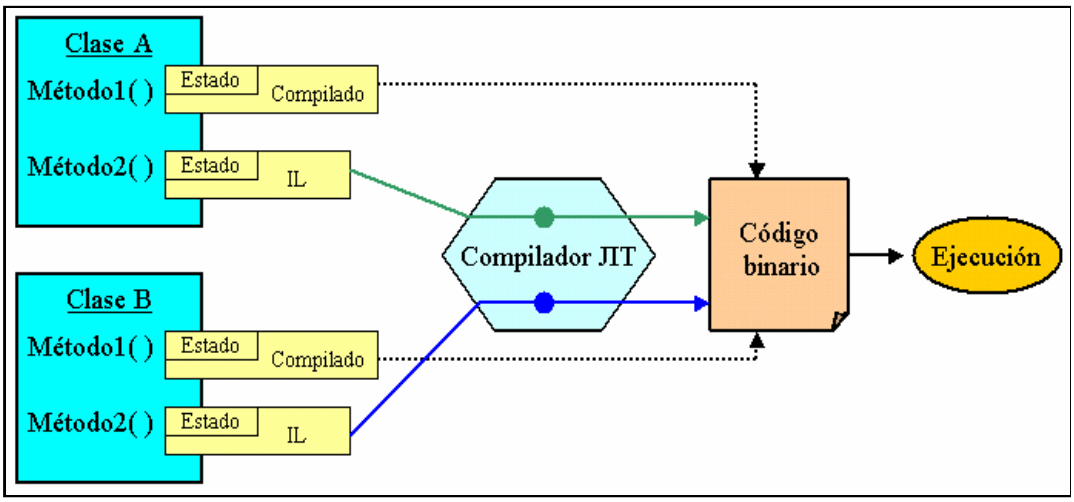
Na compilação final é efetuada toda a otimização de código como, por exemplo, trechos de código duplicados ou partes do código que nunca seriam executadas. Por este motivo no momento de executar uma MSIL não se compila o código completo, mas sim

apenas a parte que será necessária para a solicitação daquele momento, e vai armazenando todo o código caso este seja necessário em segundas chamadas.

A carga da aplicação é feita pela CLR que efetua toda a verificação de tipos e verifica o estado de cada um dos métodos e atributos e marca cada um deles.

Segundo Blanco (2002), em uma primeira chamada, a partir da identificação de estados a CLR primeiramente verifica se este método já não está compilado, e caso não esteja ele passa o controle a JIT, o qual efetua a compilação final deste para execução e modifica a marcação de estado deste método, onde para as próximas chamadas ele já estará compilado e pronto para executar.

Esta técnica de compilação pode ser vista na figura 6, conforme descrito.



Fonte: Blanco (2002)

Figura 6 - Técnica de compilação e execução da .NET

A interoperabilidade de plataformas é garantida pelo fato de existir um JIT para cada plataforma compatível de execução. A .NET oferece várias plataformas para compilação, sendo assim, um aplicativo escrito apenas uma vez pode rodar nas mais diferentes plataformas.

3.6 COMMON LANGUAGE RUNTIME

Em torno da execução comum das linguagens da .NET, a *Common Language Runtime* (CLR), representa a alma da *.NET Framework* a qual é encarregada de executar o código das aplicações escritas.

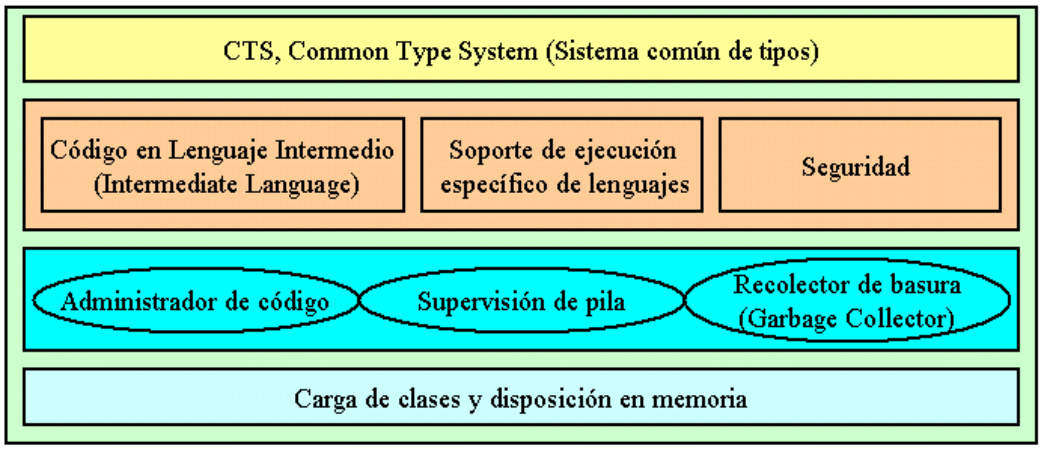
Ela possui várias características importantes como proporcionar um desenvolvimento rápido e preciso de aplicações pelo fato de grande parte das funcionalidades que normalmente os programadores deveriam se preocupar virem implementadas diretamente na execução.

Outra característica marcante é o controle da aplicação em tempo de execução gerenciando toda a carga e alocação de memória, e efetuando a coleta de lixos dos dados não mais utilizados em memória.

Oferece um sistema comum de tipos para todas as linguagens e gerencia a segurança do código que está sendo executado garantindo assim a integridade e segurança da aplicação.

> A facilidade de distribuição e liberação de aplicativos também é vista com vantagem, pois aplicações escritas podem ser facilmente colocadas em produção apenas com a copia dos arquivos que compõem a aplicação dentro da sua pasta de execução eliminando os problemas com conflito de versões e conflito entre as temidas DLL's que apresentavam os conhecidos e antigos problemas de registro e conflito (BLANCO, 2002, p. 32).

Vários são os itens que compõem a CLR conforme demonstrado na figura 7 .



Fonte: Blanco (2002)

Figura 7 - Composição da *Common Language Runtime*

3.7 COMMON TYPE SYSTEM

Constitui o mecanismo da CLR responsável pela definição dos tipos que serão manipulados e criados em torno da execução da *.NET Framework* (BLANCO, 2002, p36).

Entre as principais funcionalidades pode-se destacar a integração de códigos escritos em diferentes linguagens, otimização de código de execução, um modelo de tipos orientado a objetos que suporte múltiplas linguagens, e uma série de normas e regras que assegurem a intercomunicação entre os objetos.

Um *Common Type System* (CTS), permite que se descreva como a aplicação será executada, mas não se encarrega diretamente de sua execução, mas determina a CLR como o código deverá ser executado.

Um exemplo prático desta funcionalidade seria o fato de poder desenvolver DLL`s em C# ou J# ou ainda C++ e fazer chamadas a partir de um sistema desenvolvido em VB.Net às DLL`s desenvolvidas em outras linguagens e garantir a estabilidade e compatibilidade entre todas as linguagens utilizadas sem necessitar de conversões de tipos ou de parâmetros entre as linguagens.

3.8 COMMON LANGUAGE SPECIFICATION

Com todas as especificações a respeito da *.NET Framework* ainda resta uma pergunta não respondida: como integrar linguagens de diferentes sintaxes e fundamentos em um ambiente único?

Esta solução está contida dentro da *Common Language Specification* (CLS), onde ela consiste em um conjunto de características comuns que devem ser cumpridas por todas as linguagens integrantes da plataforma .NET para que seja possível a interação entre elas.

As principais vantagens e necessidades podem ser enumeradas como:

a) independência de linguagem, onde muitos programadores às vezes se vêem obrigados a desenvolver em uma linguagem que não é de seu domínio ou até mesmo de seu agrado, porém com a .NET isto não ocorre, pois a plataforma fornece a oportunidade de integração entre diferentes linguagens, sendo assim uma aplicação pode ser desenvolvida em mais de uma linguagem, se transformando em algo mais cômodo e prático;
b) integração de linguagens, pois é possível escrever, por exemplo, classe em uma linguagem e utilizar esta mesma classe em outros sistemas de linguagem distinta desde que ambas estejam cumprindo a CLS. Pode-se por exemplo tem

as bibliotecas desenvolvidas em C++ e estar sendo utilizada dentro de um sistema em VB, pois como as duas estão padronizadas dentro da CLS, fica transparente a utilização da mesma;

c) Abertura a novas linguagens, onde a cada dia, novas linguagens estão entrando nos padrões pré-estabelecidos e cada vez mais existem linguagens integradas com a .NET. Fabricantes terceiros já estão padronizando suas linguagens e liberando-as totalmente integradas à .NET como, por exemplo, o Cobol, o Delphi, Smaltalk entre outros.

3.9 .NET COMPACT FRAMEWORK

A *.NET Compact Framework* é um subconjunto da *.NET Framework* o qual consta de uma API totalmente orientada a objetos que permite o uso de *namespaces*, facilitando assim a organização dos elementos da API, ajudando assim na procura de novas facilidades da API.

É a plataforma de desenvolvimento para *Smart Device* da iniciativa *Microsoft .NET* e a chave para atingir clientes Microsoft com grandes experiências com o conceito de a qualquer hora, em qualquer lugar e em qualquer dispositivo.(HADDAD, 2004).

Em função de ser um subconjunto da *.NET Framework*, os desenvolvedores podem utilizar a mesma experiência e conhecimentos que possuem em programação e os códigos existentes de sistemas já produzidos através de dispositivos *desktop* e servidores. Contido dentro do *Visual Studio 2003*, a *Smart Device Application*, a qual contém o *.NET Compact Framework*, já está em produção sendo possível desenvolver qualquer tipo de aplicação para dispositivos móveis que rode a *.NET Compact Framework*.

Sendo assim com o crescente mercado dos dispositivos móveis e a exponencial demanda nos últimos tempos, agora o desenvolvimento de aplicações para este ambiente passa a ser simplesmente mais um processo de criação de *software* incluindo um alto acréscimo de eficiência e um baixo custo de produção ajudando o desenvolvimento para dispositivos móveis.

A *.NET Compact Framework* possibilita empregar o mesmo modelo de programação em camadas das aplicações cliente/servidor, e previsão para rodar em múltiplos aparelhos. Sendo assim os códigos para estas aplicações como uma lógica de negócio, camada de acesso

a dados e uma camada XML *Web Service* podem ser compartilhadas por múltiplos aparelhos e computadores. Isto aumenta consideravelmente a eficiência no desenvolvimento das aplicações.

Outro ponto importante é a robustez e a segurança na execução dos códigos desenvolvidos dentro da *.NET Compact Framework*, pois ela oferece o sistema de gerenciamento de código (*managed code*).

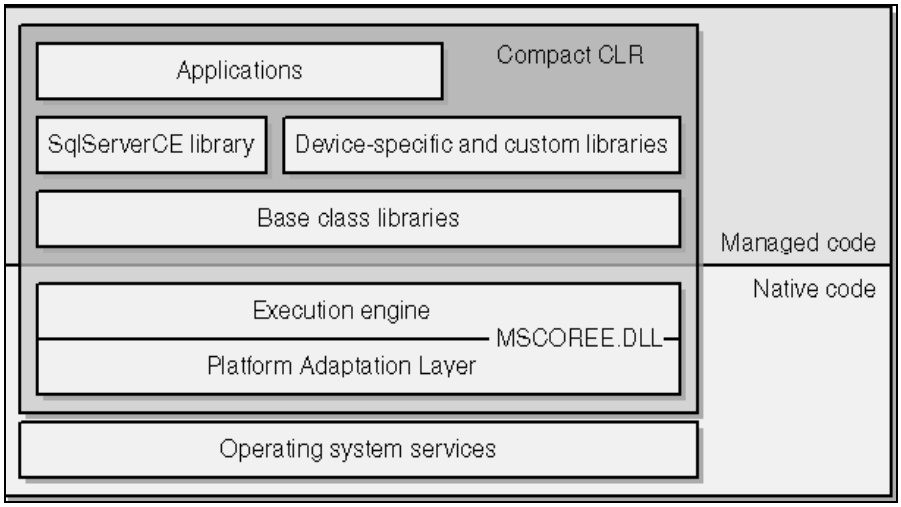
Com isso obtém-se uma maior confiabilidade no código podendo reduzir drasticamente os erros de *software*.

Um código gerenciado garante ao sistema que códigos ou problemas ocorridos no desenvolvimento travem o aparelho em tempo de execução e ao mesmo tempo não permite que códigos maliciosos obtenham acesso aos recursos de segurança do sistema.

O modelo de segurança ainda prevê *softwares* em redes distribuídas utilizando redes sem fio dentro de um caminho seguro, diminuindo a possibilidade de custo de retorno do dispositivo.

Outro ponto que deve ser destacado é a alta performance da *.NET Compact Framework*, pois ela é projetada para trabalhar com recursos limitados, normalmente encontrados em dispositivos de pequena capacidade. A eficiência da própria *.NET Compact Framework* aproveita cuidadosamente os recursos disponíveis sem desperdiçá-los.

O modelo de execução da *.NET Compact Framework* pode ser visto na figura 8.



Fonte: Wigley,Wheelwright (2003)

Figura 8 - Modelo de Execução da *.NET Compact Framework*

A *.NET Compact Framework* suporta o desenvolvimento em C#, VB.NET, C++ e J# além de que no futuro irá suportar outras linguagens devido ao *Common Language Interface* (CLI).

A .NET *CF* visa aparelhos móveis, tais como telefones móveis, *smart phones*, PDA's, *Pocket* PC's e outros componentes portáteis, como TV e eletrodomésticos, por exemplo. Atualmente só permite desenvolver aplicações para aparelhos que executam o sistema operacional *Windows CE*.

4 DISPOSITIVOS MÓVEIS

Mobilidade é o termo utilizado para identificar dispositivos que podem ser operados a distância ou sem fio. Dispositivos estes que podem ser desde um simples BIP, até os mais modernos *Pockets*.

> O início da história dos dispositivos móveis pode ter seu marco em janeiro de 1992, quando o CIO da Apple John Sculley falava sobre o novo sonho de consumo da época. O Free Online Ditionary of Computing descreve o PDA como um pequeno computador de mão utilizado para escrever notas, listar apontamentos e auxiliar na organização pessoal. (WIGLEY; WHEELWRIGHT, 2003).

Fonte: Wigley,Wheelwright (2003)
Figura 9 - Exemplo de um PDA Apple de 1992

Sensível à oportunidade do mercado que estava sendo aberto, a Microsoft em 1996 lança seu primeiro sistema operacional para dispositivos móveis, chamado de Windows CE, onde ela criou um consórcio com 40 companhias para fabricarem dispositivos compatíveis com sua nova plataforma. Porém não teve muito sucesso, pois sua interface era muito complicada e consumia muitos recursos dos dispositivos, os quais eram limitados na época.

Em 2000, com o lançamento dos Pockets PC 2000, é lançada juntamente a versão 3 do Windows CE, a qual chega bem parecida com o ambiente Windows desktop, agora mais bem elaborada e preparada para trabalhar com dispositivos móveis (WIGLEY; WHEELWRIGHT, 2003).

Mais uma atualização do sistema operacional garantiu o sucesso do Windows CE, nos iPaq Compaq que firmariam a presença da Microsoft no mundo dos dispositivos móveis.

Atualmente o Windows CE é o sistema operacional utilizado nos Pockets PC iPaq da Compaq, os modelos Jornada da HP, Axim Pocket da Dell, Pocket Loox da Fujitsu, o lançamento Maestro da Audiovox, vários modelos Toshiba como os E400, modelos Hitashi como o HPW200EC, tendo ainda outros modelos entre outros fabricantes.

Na categoria de celulares, ou Smartphones como são chamados, que suportam o sistema operacional Windows CE .NET há os modelos Nexio, SPH i500 e i700 da Samsung, Kyocera modelos QCP6035 e 7135, Siemens SX56, Motorola MPx, Nokia 9500, Sony Ericsson p800 e dependendo da região ainda há outros modelos. No Brasil apenas temos lançado comercialmente pelas operadoras o Gradiente Partner e o Motorola MPX220, mas com preços alcançáveis por poucos.

Fonte: Wigley,Wheelwright (2003)

Figura 10 - Modernos Pockets PC com a Plataforma Windows CE

Existem hoje pessoas que trabalham em ambientes empresariais não tradicional.Tal ambiente pode ser uma série de locais de trabalho temporários, como no caso dos representantes de vendas ou dos executivos em viagem, ou pode ser que a natureza de uma tarefa exija que o trabalhador esteja sempre se deslocando, como no trabalho em fábricas, entregas de pacotes, serviços de campo ou profissões ligadas à saúde.

Nos últimos cinco anos, vários aparelhos portáteis de informação, de *PC's* manuais a *Pocket PCs*, foram disponibilizados para auxiliar essa força de trabalho móvel. Esses

dispositivos não apenas ajudam no gerenciamento de compromissos e contatos como também representam uma ferramenta para substituição dos processos comerciais feitos em papel por aplicativos baseados em formulários.

Uma maior eficácia e precisão na captura rápida de dados em um dispositivo computacional pode resultar em maior produtividade dos funcionários, maior rapidez na geração de relatórios comerciais para tomadas de decisão, maior satisfação do cliente e custos operacionais reduzidos (por dispensar a entrada de dados por pessoal específico).

Segundo Depiné (2003), os dispositivos sem fio oferecem uma conectividade que outros dispositivos não possuem. Em poucos anos o desenvolvimento de aplicações para esses equipamentos tende a aumentar drasticamente, utilizando-se dos recursos que os mesmos têm a oferecer.

4.1 VANTAGENS DOS DISPOSITIVOS MÓVEIS

Muitos trabalhos hoje em dia exigem do profissional uma elasticidade e uma versatilidade que nunca se viu anteriormente. E para auxiliar estes profissionais com elementos da área de TI, eles precisam dispor dos mais modernos aparelhos portáteis existentes no mercado

Segundo Schaefer (2004, p. 21), do ponto de vista empresarial, os dispositivos móveis são ótimos geradores de informação, podendo ser utilizados na automatização do processo até nas coletas de informações estratégicas, pois com suas reduzidas dimensões podem estar sendo transportados e estar presentes em todas as situações que um profissional dessa área pode atuar.

Para estes dispositivos, cada vez mais surgem novos aplicativos exclusivos para o ambiente, trazendo ele cada vez mais para um cenário que antes era dominado por *desktops* e *notebooks*. Junto a isto cada vez mais os usuários estão usufruindo as facilidades de um mundo interligado por redes sem fio onde o usuário pode ter a qualquer hora e em qualquer lugar, a informação que desejar bastando estar apenas conectado a estas redes.

5 DESENVOLVIMENTO DO TRABALHO

Um protótipo de aplicativo rodando em um Pocket PC e um Web Service para disponibilizar a sincronização de dados formam o aplicativo desenvolvido, tendo como objetivo fornecer informações comerciais referentes aos clientes de um representante ou vendedor, em um equipamento portátil e de fácil manuseio e ainda efetuar a partir deste mesmo dispositivo móvel o envio de pedidos a uma base de dados possibilitando recepção e processamento dos pedidos com a possibilidade de integração com um outro sistema comercial.

O aplicativo desenvolvido para o Pocket PC é a principal parte do aplicativo e o principal foco do desenvolvimento, sendo executado e testado em um emulador contido dentro da ferramenta de desenvolvimento Visual Studio.NET, desenvolvido para a plataforma .NET para rodarem em dispositivos que suportem a Framework.NET e o Windows CE .NET 4.2.

Há também um protótipo de Web Service, simulando a atualização de dados a partir de um repositório de dados centralizado que fornecerá as informações para alimentar a base de dados local do Pocket PC e que também receberá as informações de pedidos enviados durante a transmissão e recepção de dados e gravará no repositório centralizado.

Neste trabalho é implementado um sistema CRM para dispositivos móveis desenvolvido especialmente para usuários focados no atendimento a clientes citando como exemplos vendedores e representantes, que trabalham a maior parte do tempo em viagens e atendimentos *in loco* ao cliente e que necessitam de alta mobilidade e praticidade em suas ferramentas que auxiliam no trabalho de campo e que ao mesmo tempo precisam associar isto à necessidade de possuir informações atualizadas em tempo real.

Com isto, será possível efetuar negociações com clientes, pois tendo as informações do cliente a mão facilita e agiliza tomada de decisões estratégicas e fornece um maior dinamismo e liberdade ao representante comercial, pois as informações contidas no dispositivo dispensam qualquer contato com a unidade base para se conseguir informações que auxiliem no conhecimento do cliente para desenvolvimento de relacionamento com os clientes e também, poder enviar pedidos diretamente a base central com toda a segurança.

As informações que estão sendo apresentadas neste sistema foram coletadas com 2 analistas de marketing de uma multinacional de grande porte da região de Jaraguá do Sul e também alguns representantes comerciais desta mesma empresa, onde, depois do levantamento, tudo foi condensado em um misto entre as sugestões e foi desenvolvida a técnica que levará informações necessárias a serem gerenciadas pelo sistema.

Com certeza este sistema trata genericamente a parte de força de vendas e informações comerciais, sistema este que se enquadra na categoria do CRM operacional, que é coletar e trabalhar com as informações diretamente com o cliente podendo ser enquadrado em vários segmentos empresariais e ainda ser integrado com diversos sistemas comerciais.

5.1 REQUISITOS LEVANTADOS DO PROBLEMA A SER TRABALHADO

A seguir são descritos os requisitos funcionais, não funcionais e as regras de negócio levantadas para a implementação do trabalho:

a) RF1 – Consultar Clientes: as informações de clientes poderão ser acessadas partindo de uma base de dados local importada e atualizada. Os dados de clientes serão somente para leitura, não possuindo possibilidade de cadastramento e implantação de clientes. Todos serão provenientes de uma base de dados importada a partir de outro sistema que mantém os clientes. A consulta poderá ser feita pelo nome do cliente ou código do cliente. No resultado será apresentada uma lista onde clicando sobre o cliente desejado abrirá um breve cadastro com informações básicas com nome da empresa, código da empresa, CNPJ, endereço, status cadastral (ativo ou inativo), colocação de mais valoroso cliente e telefone;

   RF1.1 – Manter Contatos de Clientes: o sistema deverá disponibilizar para o usuário a possibilidade de cadastramento e manutenção de contatos de um cliente. Deverá dispor de nome, telefone para contato comercial, data de nascimento, sexo, estado civil e observações;

b) RF2 – Consultar Estoques: os estoques conterão a posição de estoque conforme a última atualização do sistema, possuindo também vendas efetuadas para este item e também os planejamentos de produção dos próximos 60 dias. Será uma base apenas para consulta, não contendo alteração, inclusão ou exclusão de dados. A pesquisa de itens para consulta poderá ser feita por código ou informando parte do nome do produto a ser buscado em outra tela específica para pesquisa de itens;

RF2.1 – Consulta Preço do Item: juntamente com a consulta de estoque, deverá na mesma tela aparecer o preço do item solicitado, sem considerar nenhum cálculo de descontos (Preço Bruto);

c) RF3 – Consulta Notas Fiscais: deverão ser apresentadas as notas emitidas para o cliente referente aos últimos 3 meses, ou as notas que estiverem em aberto e ultrapassarem este limite. Deverá ser apresentada informação básica da capa da nota, como número da NF, data de emissão, data de saída, local de entrega, condição de entrega e transportadora. A consulta poderá ser feita por código do cliente informando ainda a data de emissão da nota, ou apenas pela data de emissão da nota para saber todas as notas emitidas para o representante;

   RF3.1 – Itens da Nota Fiscal: deverão informar dados básicos como código de item, descrição de item, quantidade, valor faturado unitário e total e código do pedido caso tenha mais de um pedido na mesma nota fiscal e também controle de seqüência dos itens;

d) RF4 – Consulta Duplicatas em aberto: as duplicatas que ainda não foram pagas pelo cliente deverão ser todas apresentadas. As duplicatas em atraso deverão trazer o cálculo dos juros do atraso. A consulta pode ser feita pelo código do cliente apresentando todas as suas duplicatas, ou filtrar por data de emissão da duplicata, ou apenas apresentar a data para verificar todas as duplicatas do representante;

e) RF5 – Manter Pedidos: poderão ser feitas emissões e consultas de pedidos para os clientes. A capa do pedido deverá possuir informações básicas do pedido como número do pedido, número do pedido do cliente, data de emissão, informações do local de entrega e tipo de frete, condição de pagamento e observações de pedido;

   RF5.1 – Itens do Pedido: os itens do pedido deverão ser listados em uma segunda tela apresentando o código do item, descrição do item, seqüência , quantidade, data de entrega planejada, valor unitário e total do item;

   RF5.1.1 – Inclusão e Exclusão de Itens: a inclusão se dará por meio de código direto na tela de Itens do Pedido ou poderá ser feita uma breve pesquisa em uma outra tela para este destino. A tela retornará os resultados que atendem o requisito de pesquisa e disponibilizará um *link* no item para ser clicado e automaticamente inserido no pedido, bastando depois informar apenas a quantidade. A exclusão será feita por meio de marcação do item e um botão específico para exclusão de itens;

RF5.2 – Consulta Pedidos: o sistema deverá possuir uma tela para efetuar pesquisa de pedidos emitidos para o cliente nos últimos 6 meses. A pesquisa poderá ser feita por intervalo de data, ou pelo código do pedido, tanto do cliente como da empresa. Será apresentado na tela de resultado da pesquisa, o código do pedido, data de emissão e valor total do pedido. Clicando ele abrirá o pedido completo e seus itens;

RF5.3 – Referência do Pedido: o sistema deverá apresentar automaticamente o número de referência do pedido do representante. Numeração seqüencial;

f) RF6 – Histórico do Cliente: o histórico traça um perfil médio do cliente, fornecendo informações importantes como compras nos últimos 6 meses, compras nos últimos dois anos, limite de crédito do cliente, e atraso médio de pagamento de duplicatas. Fornece também os pagamentos efetuados nos últimos 6 meses. A pesquisa deve ser feita sempre pelo código do cliente;

RF6.1 – Compras dos últimos 6 meses: as compras dos últimos seis meses são os valores de pedidos emitidos a este cliente nos últimos seis meses, contando a data da última atualização de sistema efetuada pelo usuário. Deverá estar separada mês a mês e mostrar o totalizador no final da soma;

RF6.2 – Compras dos últimos 2 anos: as compras dos últimos dois anos são os valores de pedidos emitidos pelo cliente nos últimos dois anos, contando a data da última atualização efetuada pelo usuário. Deverá ser separada ano a ano e possuir um totalizador no final da soma dos anos;

RF6.3 – Limite de Crédito: será informado o limite de crédito do cliente e o valor disponível do limite;

RF6.4 – Pagamentos Efetuados: refere-se às duplicatas pagas pelo cliente referentes aos seus faturamentos. Deverá ser mostrado o pagamento total separado mês a mês e na mesma linha apresentar um valor de atraso médio de pagamento de duplicatas;

g) RF7 – Sincronização de Dados: o envio e recebimento de informações referente aos negócios deverão ser feitos via um sistema *online* de troca de informações, onde a qualquer momento, o representante, estando conectado a uma rede com acesso a internet, possa atualizar as informações do dispositivo móvel. Devera ser feita apenas a atualização dos dados que contenham alguma alteração para otimizar a transferência de dados;

h) NEG1 – Clientes Apresentados na Pesquisa de Clientes: serão apresentados todos os clientes do representante, com todos os status cadastrais, ou seja, tanto os ativos,

como inativos e bloqueados serão apresentados na listagem dos clientes do representante;

i) NEG2 – Apresentação do Estoque e Preços: os estoques apresentados se referem à posição instantânea e os planejamentos de produção se referem apenas ao período firme, ou seja, que já possuem confirmação de produção. Os preços de itens serão apresentados de uma tabela de preços bruta sem nenhum tipo de desconto ou abatimento;

j) NEG3 – Duplicatas em Aberto: as duplicatas em aberto compreendem aquelas que foram emitidas a partir de uma nota fiscal, porém ainda não foram liquidadas pelo cliente. Devem ser apresentadas as duplicatas vencidas e que estão a vencer ;

k) NEG4 – Emissão de pedidos: os pedidos possuirão um número seqüencial mais uma cadeia de 3 caracteres concatenados para designar a numeração de pedido do representante. Os pedidos depois de emitidos e constando apenas na base local do *pocket pc* ainda podem ser modificados sem nenhum problema. Depois de enviados para a base central de recepção o pedido ficará com *status* bloqueado, não podendo mais ser modificado;

l) NEG5 – Histórico do Cliente: o histórico do cliente deve considerar também as devoluções e não considerar notas de simples remessa. A média de atraso de duplicatas apresentada nessa tela é o total de dias de atraso dividido pelo número de duplicatas atrasadas do cliente;

m) RNF1 – Mobilidade: o sistema deverá estar preparado para rodar em aparelhos tipo Pocket PC, com atualização de dados via *web service* e base de dados *stand alone*;

n) RNF2 – Interface: a interface deve prever as limitações de um aparelho portátil, trabalhando com telas paginadas e com facilidade de trabalhar apenas com ponteiros, devendo ter o mínimo de campos para preenchimento manual;

o) RNF3 – Base de Dados: a base de dados deve ser otimizada para que os dados não ocupem um grande espaço na base. Deve ter um limite de dados de até 16MB a ser trazido para o aparelho;

p) RNF4 – Velocidade: os dados devem estar armazenados de forma mais correta possível e as pesquisas e operações de banco de dados altamente otimizadas garantindo performance ao equipamento. Deve possuir rápida resposta aos comandos por se tratar de um sistema a ser utilizando no *front* de trabalho;

5.2 ESPECIFICAÇÃO

Na especificação técnica do sistema, estaremos apresentando modelos baseados na linguagem UML, descrevendo os diagramas de classe, diagramas de caso de uso, diagrama de atividade e diagrama de seqüência do caso primário do sistema. Estará ainda sendo especificado textualmente as funções do sistema em seus principais módulos.

5.2.1 Diagrama de Caso de Uso

Apresenta-se os casos de uso primário e secundário a seguir, explicando de maneira mais detalhada os casos de maior influência dentro do sistema.

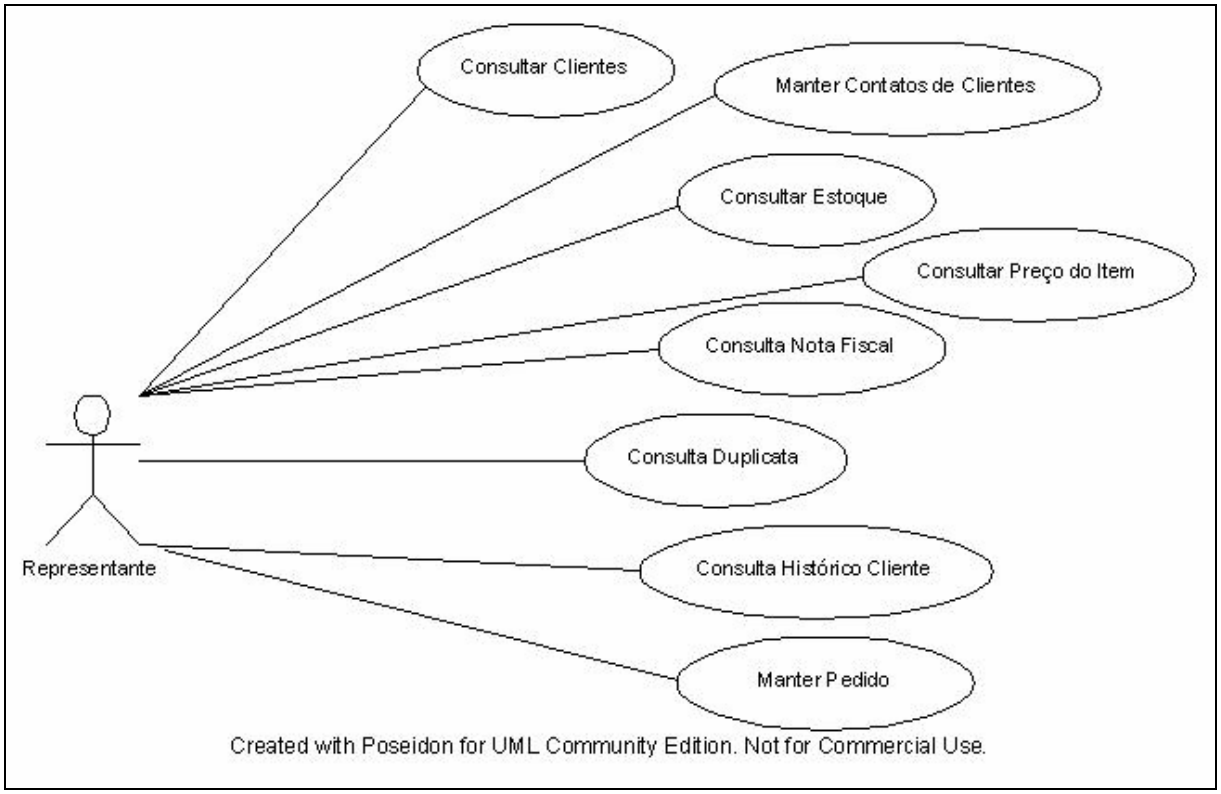


Figura 11: Diagrama de caso de uso primário

5.2.1.1 Manter Pedidos

O representante comercial poderá inserir, excluir e atualizar pedidos a partir da funcionalidade disponível no sistema (figura 11). O sistema contará com um módulo de pedidos, onde serão feitas pesquisas de cliente a partir do código do cliente, que é informado logo na entrada do sistema, ou pode ser informado no campo disponível para código do cliente e depois clicando no botão de disparo para a pesquisa na própria tela de pedidos. A

mesma pesquisa contará com dois campos para informar um intervalo de datas e uma grade que trará todas os pedidos que atenderem ao filtro solicitado. Serão três botões: um para disparar a pesquisa, um para excluir o pedido e outro para iniciar a inclusão de um novo pedido. No caso da pesquisa, ele utilizará os valores informados na tela e trará o código, data e valor do pedido no grade para ser selecionado. No caso de nova inclusão, após clicar no botão ele enviará diretamente para a tela de cadastramento de pedidos, carregando por padrão as informações já cadastradas do cliente. Depois de conferidos os dados da capa do pedido, clicando no botão gravar, ele irá gravar a capa do pedido, criará um novo número de pedido e em seguida encaminhará diretamente para a tela de itens do pedido. A tela de itens do pedido possuirá um campo para informar o código do produto, um campo para mostrar a descrição do produto, campo este que não será editável, um campo para informar a quantidade, outro para apresentar o valor unitário e um último para apresentar o valor total. O campo valor unitário e quantidade serão editáveis. Depois de gravado o pedido, ele ficará a disposição do sistema de sincronização de dados, onde, depois de efetuada esta sincronização, o pedido passa a ser somente leitura e ficará apenas disponível para consulta. No caso da exclusão, só poderão ser excluídos pedidos que ainda não foram enviados via sistema de sincronismo para o repositório central. No menu de navegação da tela possuirá um botão que encaminhará para a pesquisa de cliente, um para a capa do pedido e outro para o menu principal.

Quadro 1: Caso de Uso Inserir Novo Pedido e Itens

Caso Uso 1 : Inserir Novo Pedido e Itens	
Importância	Alta
Ator primário	Usuário Representante
Ator secundário	Não há
Pré-condições	Representante deve ter acesso ao sistema

Fluxo principal	- Representante acessa o sistema e seleciona o cliente
	- Representante recebe do cliente os itens a serem incluídos no pedido
	- Representante cadastra os itens no pedido
	  - Para cada item a ser incluído
	  - Seleciona Item
	  - Verifica Preços
	  - Verifica quantidade
	- Representante salva o pedido
	- Representante prepara conexão para envio de pedido

Fluxo alternativo	- Caso não exista o cliente cadastrado
	  - Cadastrar cliente como status de Prospect
	  - Cadastrar contatos do cliente
	- Caso não possua conexão para envio
	  - Salvar pedido para envio posterior

Pós-condições	Pedido Cadastrado e Implantado
Regras do negócio	Cliente e contato deverão estar devidamente cadastrados e importados para o sistema Itens deverão estar devidamente cadastrados e importados para o sistema Quantidades deverão respeitar os múltiplos dos itens.

5.2.1.2 Consultar Clientes

A base de clientes ficará disponível apenas para consultas, onde as formas de pesquisa são pelo nome do cliente, informado completamente ou apenas parte do nome, ou pelo código do cliente. Ambos os resultados serão carregados em uma grade que informará código de cliente, nome de cliente, cidade e estado do cliente e que com dois cliques sobre a linha do cliente, será encaminhado para a tela de detalhes do cliente que virá preenchida com todas as informações do cliente, trazendo os campos código, nome, cadastro nacional de pessoa jurídica (CNPJ), endereço, bairro, cidade, estado, situação cadastral, o qual consta qual a situação do cliente como, por exemplo, ativo, inativo, bloqueado por credito e outros, telefone e colocação de mais valoroso cliente (MVC) que é a colocação entre os mais valorosos clientes da carteira da empresa. Sendo assim a tela possuirá um campo para descrever o nome ou parte do nome do cliente, um campo para código do cliente e um botão “Pesquisar” para disparar o processo de consulta. Existirá uma grade para trazer em suas linhas os resultados encontrados na base. As informações de clientes são apenas leitura, não possibilitando modificações. Na mesma tela, no menu de navegação existirá botão para acesso aos detalhes do cliente, contatos do cliente e menu principal.

Quadro 2: Caso de Uso Pesquisar Cliente

Caso Uso 2 : Pesquisar Cliente	
Importância	Alta
Ator primário	Usuário Representante
Ator secundário	Não há
Pré-condições	Representante deve ter acesso ao sistema Representante deve possuir cliente cadastrado em carteira Cliente tem que estar ativo
Fluxo principal	- Representante acessa o sistema e seleciona tela de consulta de cliente - Representante informa no campo ou parte do nome ou o código do cliente - Representante verifica resultados obtidos a partir da consulta na grade de consulta - Representante detalha informações do cliente
Fluxo alternativo	- Caso não exista o cliente cadastrado - Cadastrar cliente como Prospect - Cadastrar contatos do cliente
Pós-condições	Informações cadastrais do cliente obtidas
Regras do negócio	Cliente e contato deverão estar devidamente cadastrados e importados para o sistema Cliente deverá pertencer a carteira do representante.

5.2.1.3 Manter Contatos de Clientes

Os contatos de clientes serão livres para pesquisa, inclusão, exclusão e manutenção dentro do sistema. Na tela de pesquisa de contatos, a partir do código do cliente informado na tela de acesso ao sistema, por parte do nome, ou mesmo deixando o campo de filtro do nome em branco, serão apresentados todos os contatos do cliente na grade da tela, com o código do

contato, nome e telefone comercial. Com um duplo clique sobre a linha da grade, poderão ser acessados os dados de detalhe do contato do cliente. Os contatos estarão associados diretamente a um cliente, e seu cadastro constará de código do contato, nome, telefone comercial, data de nascimento, estado civil e um campo para observações gerais. Estas informações também serão enviadas via sistema de sincronização de dados para a empresa, onde a cada modificação o item ficará marcado para efetuar a atualização. No menu de navegação poderá ser feito acesso direto à tela de informações de histórico de dois anos/limites, pagamentos e voltar ao menu principal.

Quadro 3: Caso de Uso Manter Contato Cliente

Caso Uso 3 : Manter Contato Cliente	
Importância	Baixa
Ator primário	Usuário Representante
Ator secundário	Não há
Pré-condições	Representante deve ter acesso ao sistema Representante deve possuir cliente cadastrado em carteira Cliente tem que estar ativo
Fluxo principal	- Representante acessa o sistema e seleciona tela de consulta de cliente - Representante informa o cliente para o sistema - Representante acessa a tela de contatos de cliente e pesquisa os contatos a partir do filtro da tela. - Representante verifica necessidade de inclusão do contato - Caso necessite, representante inclui um novo contato e insere as informações necessárias - Representante detalha informações do contato e efetua sincronização de dados.
Fluxo alternativo	- Caso não exista o cliente cadastrado - Cadastrar cliente como Prospect - Cadastrar contatos do cliente
Pós-condições	Informações cadastrais do contato do cliente obtida Cadastramento de um novo contato de cliente
Regras do negócio	Cliente e contato deverão estar devidamente cadastrados e importados para o sistema Cliente deverá ser cliente ativo no sistema e pertencer a carteira do representante.

5.2.1.4 Consultar Estoque

Nos estoques estará disponibilizada apenas a opção de consulta, não possuindo nenhum tipo de manutenção de informação. Sendo assim a consulta de estoque é apenas leitura. A pesquisa é feita a partir do código de item de produto onde, depois de informado e clicado no botão pesquisar, ele irá trazer a descrição do produto, a unidade de medida, quantidade, preço e um grade apresentando todas as programações de vendas e produção do item. No menu de navegação possuirá um botão para pesquisa de item, e um botão para o menu principal.

Quadro 4: Caso de Uso Consultar Estoque

Caso Uso 4 : Consultar Estoque	
Importância	Baixa
Ator primário	Usuário Representante
Ator secundário	Não há
Pré-condições	Representante deve ter acesso ao sistema
Fluxo principal	- Representante acessa o sistema e seleciona tela de consulta de estoque - Representante informa o código do item - Após a pesquisa, analisa os dados retornados..
Fluxo alternativo	- Caso item não cadastrado, efetuar sincronização de dados
Pós-condições	Obtenção de informações sobre estoque e programação de itens
Regras do negócio	Item deve estar cadastrado e importado para o sistema.

5.2.1.5 Consultar Preço do Item

O preço do item está diretamente ligado ao item sendo apenas consulta. A tela de itens contará também com a informação de embalagem e unidade de medida, caso o item possua mais que uma unidade de medida. O preço do item será trazido juntamente com a consulta de estoques, em um campo definido para este fim.

5.2.1.6 Consultar Notas Fiscais

As notas fiscais serão consultadas a partir das informações atualizadas do repositório central de dados. As informações estarão atualizadas até a última sincronização de dados. São informações de somente leitura, possuindo na tela de filtro para pesquisa um campo para o número da nota, outro para a série da nota, campo para informação de cliente e dois campos para informar um intervalo de data de emissão da nota. Ao clicar em pesquisar, abaixo dos campos de pesquisa existe uma grade que será carregada com todas as notas fiscais que atenderem ao solicitado. Na grade existirá uma coluna com o número e série da nota, data de emissão, valor da nota fiscal, e número do pedido da nota (se existir mais de um pedido, apresentará apenas um texto informando que existe mais de um pedido para a nota). Para consultar os detalhes da nota fiscal, será clicado sobre a linha desejada e será encaminhado diretamente para a capa da nota fiscal apresentando suas informações e possuindo um botão para encaminhar para a tela de itens da nota fiscal. No menu de navegação da tela de filtro e consulta de nota fiscal constará um botão que direciona para a tela de clientes, um botão que direciona para a capa de nota fiscal e um botão que direciona para o menu principal. Na tela de capa de nota fiscal aparece um botão para a tela de consulta e filtro de nota fiscal, um botão para os itens da nota e um botão para o menu principal.

Quadro 5: Caso de Uso Consultar Notas Fiscais

Caso Uso 5 : Consultar Notas Fiscais	
Importância	Alta
Ator primário	Usuário Representante
Ator secundário	Não há
Pré-condições	Representante deve ter acesso ao sistema Representante deve possuir cliente cadastrado em carteira Cliente tem que estar ativo Cliente deve possuir pedidos emitidos Cliente deve possuir notas fiscais faturadas
Fluxo principal	- Representante acessa o sistema e seleciona tela de consulta de cliente. - Representante informa o código do cliente para o sistema. - Representante acessa a tela de consulta de notas fiscais e pesquisa as notas fiscais a partir dos filtros da tela. - Representante verifica as notas na grade e seleciona nota desejada para detalhar a capa e os itens da nota.
Fluxo alternativo	- Caso não exista nota fiscal cadastrada pro cliente, o sistema informa que não existe nota fiscal cadastrada.
Pós-condições	Informações referentes a notas fiscais do cliente apresentadas ao representante
Regras do negócio	Cliente deverá estar devidamente cadastrado e importado para o sistema. Cliente deverá ser cliente ativo no sistema e pertencer à carteira do representante.. Cliente deve possuir notas fiscais emitidas. Cliente deve possuir pedido emitido para a geração do faturamento.

5.2.1.7 Consultar Duplicatas

As informações de duplicatas serão buscadas diretamente do repositório de dados central a partir das sincronizações de dados. As informações serão somente leitura e estarão atualizadas com a posição da última sincronização efetuada.

A tela possui um campo para informação e pesquisa do código do cliente, e dois campos para informação de um intervalo de datas de pesquisa. Ao clicar em pesquisar, será carregado o grade logo abaixo dos campos de filtro com a informação do código da duplicata, valor da duplicata e data de emissão e vencimento.

Quadro6: Caso de Uso Consultar Duplicata

Caso Uso 6 : Consultar Duplicata	
Importância	Média
Ator primário	Usuário Representante
Ator secundário	Não há
Pré-condições	Representante deve ter acesso ao sistema
Fluxo principal	- Representante acessa o sistema e seleciona tela de consulta de duplicatas - Representante informa o código do cliente e o intervalo de datas de consultas. - Efetua a pesquisa e analisa os clientes com duplicatas em aberto..
Fluxo alternativo	- Caso cliente não possua duplicatas, sistema avisa que nenhuma duplicata está em aberto
Pós-condições	Obtenção de informações sobre duplicatas em aberto
Regras do negócio	Cliente deve possuir duplicatas em aberto.

5.2.1.8 Consultar Histórico do Cliente

As informações de histórico de clientes será processada a partir do repositório de dados e enviada via sincronização de dados para o representante. As informações serão somente leitura e estarão com a situação da última data da sincronização de dados. A consulta será feita a partir da tela de filtro, informando o código do cliente.

As telas possuem informações pré-definidas, onde na primeira tela serão apresentados os valores faturados para o cliente nos últimos seis meses, apresentando ao final um totalizador. Na segunda tela tem-se apresentado às informações de histórico financeiro de compras dos últimos dois anos e também apresentando os limites de crédito do cliente e o limite bloqueado. Na terceira tela, constará o histórico de pagamento de duplicatas dos últimos doze meses, e também um totalizador dos valores pagos.

No menu de navegação, irá constar na tela de valores faturados dos últimos seis meses um botão direcionando para a tela de valores dos últimos dois anos, mais um botão para a tela de pagamentos efetuados e um para o menu principal. Na tela de histórico dos últimos dois anos, terá um botão direcionando para os faturamentos mensais, outro para a tela de pagamentos e o último, para o menu principal. E por último a tela de pagamentos terá dois botões ligando às telas anteriores e um para o menu principal também.

Quadro 7: Caso de Uso Consultar Histórico Cliente

Caso Uso 6 : Consultar Histórico do Cliente	
Importância	Média
Ator primário	Usuário Representante
Ator secundário	Não há
Pré-condições	Representante deve ter acesso ao sistema
Fluxo principal	- Representante acessa o sistema e seleciona tela de consulta de Histórico de cliente - Representante entra na primeira tela de histórico e informa o cliente - Efetua a pesquisa e analisa os dados de histórico navegando nas três telas pelo menu de telas existente entre as três telas
Fluxo alternativo	- Caso cliente não possua histórico de vendas o sistema informará que o cliente não possui movimentação
Pós-condições	Obtenção de informações sobre histórico financeiro e limite de credito do cliente
Regras do negócio	Cliente deve estar cadastrado e possuir movimentação.

A figura 12 apresenta o diagrama de caso de uso secundário do sistema.

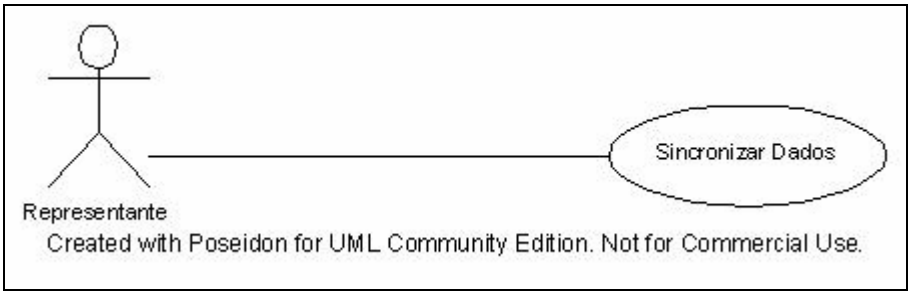


Figura 12: Diagrama de caso de uso secundário

5.2.1.9 Sincronização de Dados

A sincronização de dados será feita a partir do dispositivo, onde após se conectar a um *web service* implementado para prover as informações onde será enviado para o *web service* à solicitação de preparação do pacote de atualização e também será enviado um arquivo contendo todas as modificações efetuadas no sistema. A sincronização será feita de forma manual, a partir da opção disponível no menu inicial do sistema, onde serão varridas todas as tabelas do sistema a procura de modificações nos registros das tabelas de pedidos. No repositório central, serão varridas as informações importantes e enviadas para o dispositivo para que seja efetuada a atualização da base de dados *stand alone*.

5.2.2 Diagramas de Modelo de Classe

Para efetuar a especificação do sistema foi desenvolvido o diagrama de classes do aplicativo, orientado a objetos utilizando a linguagem *Unified Modeling Language* (UML).

O sistema é composto por vários objetos de negócio, onde cada um representa uma necessidade básica que deverá ser comportada pelo sistema controlando todas as ações e poderão ser tomadas por cada item. Dentro da implementação, a interface da aplicação apenas irá solicitar aos objetos de negócio as informações necessárias e os objetos de negócio irão devolver apenas os dados já formatados e preparados para serem apresentados na interface da aplicação.

O acesso ao banco de dados será feito a partir de uma classe implementada apenas para este fim, onde todos os métodos estão preparados para efetuarem as transações de banco de dados e devolverem apenas as informações solicitadas pela classe de negócio. Com isto, os

objetos de negócio apenas irão gerenciar os dados a serem apresentados e a busca dos dados será feita diretamente por uma classe implementada para este fim.

A figura 13 apresenta o modelo de diagrama de classes:

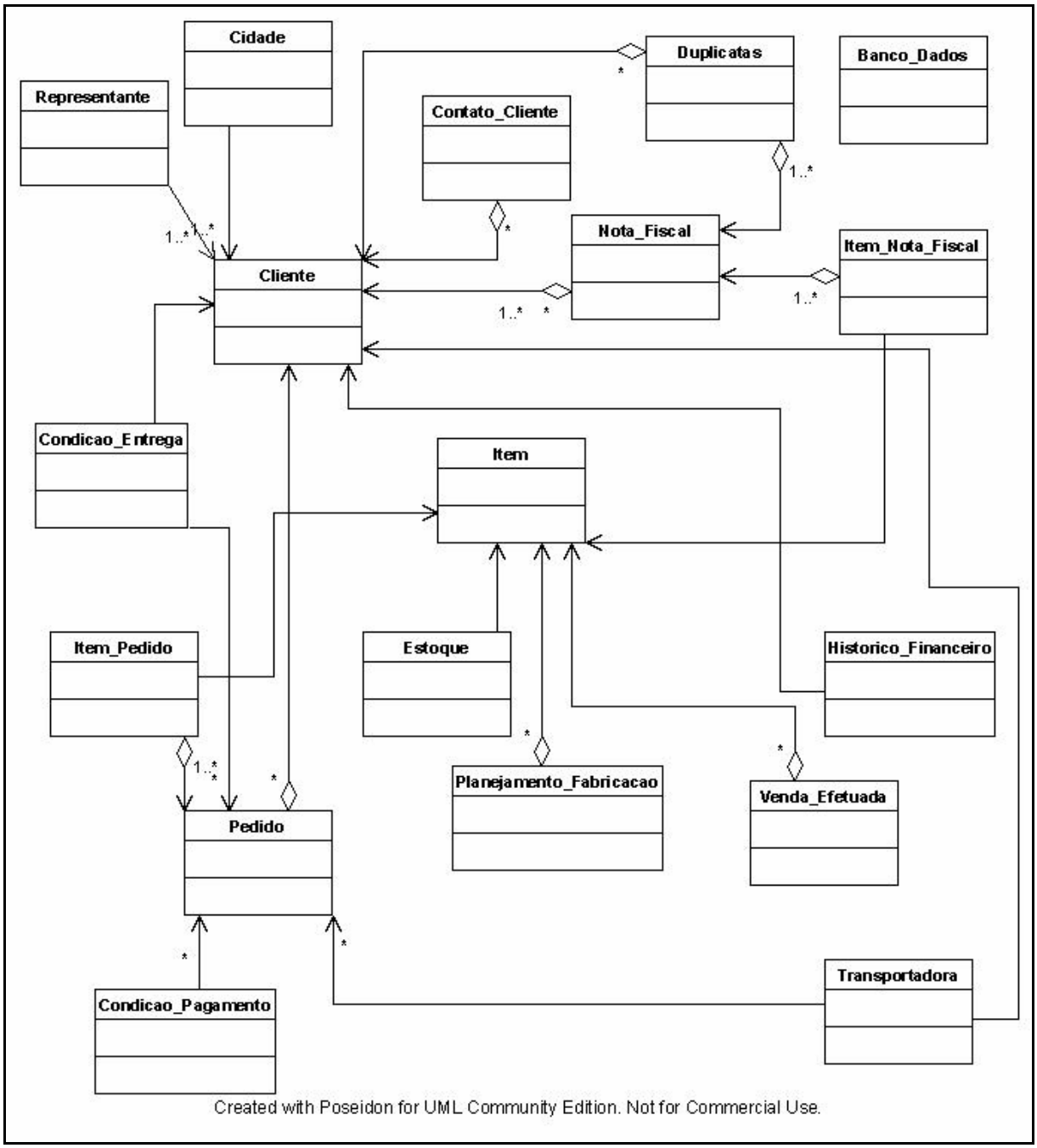


Figura 13: Diagrama de Classes

O modelo conta com 18 classes, onde 17 estarão focadas diretamente no negócio e uma classe estará disposta especialmente para prover acesso aos dados do banco de dados.

Explicando melhor as classes implementadas (Figura 13) :

a) Pedido: é a classe responsável por todos os atributos e métodos que gerenciam as possíveis transações sobre um pedido. Considerada a principal classe do sistema, pois implementa não apenas as funções de leitura das informações fornecidas pelo repositório central, mas sim troca informações com o repositório central trabalhando com o envio e processamento de informações para repositório central e também pelo processamento de informações recebidas. Os principais métodos são listados a seguir:
   - gera_CodigoPedido: não possui nenhum parâmetro de entrada, trata da geração e seqüência de código do novo pedido. A partir de uma seqüência lógica, gera o código e traz em formato *string* para atender a solicitação,
   - busca_Pedido: recebe como parâmetros de entrada o código de pedido(cdPedido), data de pesquisa inicial(dtInicial)e a data de pesquisa final (dtFinal). Com estas informações efetua a busca de pedidos, obedecendo o intervalo de datas caso ela seja informada, ou busca apenas pelo código do pedido. Dentro do método é verificado qual das informações foram passadas para efetuar a seleção dos dados e retorná-los em um *DataTable*,
   - grava_CapaPedido: recebe como parâmetro o tipo de transação a ser efetuada, se é uma atualização dos dados já existentes em um pedido, ou se é a gravação de um novo pedido. Efetua a gravação dos dados e retorna um *boolean* informando se a gravação foi concluída com sucesso ou não,
   - verifica_StatusPedido: efetua verificação no pedido, onde ele poderá estar com status de enviado ou aguardando envio. Isso se faz importante, pois antes do envio o pedido continua aberto para alterações, porém após enviado não pode mais ser modificado, ficando como apenas leitura. Retorna um *boolean* informando se está em aberto ou não;

b) Item_Pedido: é a classe responsável por gerenciar os atributos e métodos referentes aos itens do pedido. Efetua todo o controle de manutenção dos itens de um pedido, sendo a inserção, exclusão e modificação e seleção dos itens de um pedido. Considerada também uma das principais classes do sistema, pois junto com a classe de

pedidos é quem efetua o envio, recebimento e manutenção dos dados de pedidos que são trocados com o repositório central. Abaixo são explicados os principais métodos:

- calcula_ValorTotal: método responsável por efetuar o cálculo de preço de todos os itens inseridos no pedido, recebendo como parâmetros o valor unitário do item (vlUnitario) e a quantidade solicitada (vlQtdade) retornando o valor total do item,
- busca_ItensPedido: ao chamar este método será solicitada a busca dos itens de um pedido, passando como parâmetro de entrada o código do pedido (cdPedido) e recebendo como saída um *DataTable* com os itens do pedido. Serão verificadas na classe as condições de apresentar os itens e a busca em banco será feita através da classe implementada para interface com o banco de dados,
- grava_ItensPedido: este método insere novos itens no pedido, ou atualiza as informações de um item já cadastrado. É passado como parâmetro, o tipo de transação a ser efetuada(tpTipoTransacao), onde ele informará se é uma atualização ou uma inserção de dados,
- deleta_ItensPedido: quando for necessário deletar um item do pedido, será efetuada uma chamada a este método, enviando como parâmetros o código do pedido (cdPedido) e o código do item (cdItem). Será retornado um *DataTable* com a nova condição dos itens do pedido;

c) Cliente: consta dos atributos e métodos de identificação do cliente, onde todas as operações efetuadas sobre os clientes ficam implementadas. Possui a parte de negócio implementada nela, onde todas as condições comerciais são verificadas para apresentação do cliente na tela, e faz acesso a classe de Banco_Dados para executar as tarefas de banco de dados. Esta classe basicamente fará manipulação de consulta, onde os dados de clientes são apenas leitura para o sistema. Abaixo detalha-se o principal método da classe: busca_Cliente, apresentará como retorno um *DataTable* com as informações de clientes, a partir do filtro a ser preenchido na tela de consulta cliente. Método sobrecarregado, pode ser chamado passando como parâmetro o código do cliente (cdCliente) ou então passando parte do nome do cliente (nmCliente);

d) Contato_Cliente: todo cliente possui agregado a ele contatos comerciais, ou seja, pessoas de contato dentro da empresa para verificações e negociações. Os contatos serão manipulados por esta classe, onde ela possui atributos e métodos para descrever

os clientes. Ela possuirá também apenas a parte de negócio, onde poderão ser efetuadas manutenções, inclusões e exclusões de cadastros de contatos a partir desta classe. Ela fará também acesso aos dados via a classe Banco_dados, com os metodos:

- busca_ContatoCliente: efetua a pesquisa dos contatos de um cliente recebendo como parâmetro o código do cliente (cdCliente) e trazendo como retorno um *DataTable* com as informações de clientes. As informações serão buscadas em banco a partir da classe de interface com o banco de dados,
- grava_ContatoCliente: a atualização e inserção de novos contatos para um cliente é efetuado a partir deste método. Recebendo como parâmetro o tipo de transação (tpTransacaoBanco) que deverá ser executada, a classe efetua a verificação das informações e qual o tipo de transação de banco será executada. O retorno deste método é um *boolean* informando se a transação foi concluída com sucesso,
- deleta_ContatoCliente: quando for necessária a exclusão de um contato de cliente, será chamado este método responsável pelo procedimento. Ele efetuara as verificações, e caso possa ser efetuada a exclusão, ele solicitará a classe de banco de dados que execute a solicitação. Retornará um *boolean* informando a conclusão da transação com sucesso;

e) Nota_Fiscal: classe responsável pelas informações recebidas referentes a notas fiscais do repositório de dados central. Atualizada pelo sistema de sincronização de dados, é de apenas leitura de informações, não possuindo nenhum tipo de operação que não seja consulta. Possi todos os atributos necessários para apresentação das principais informações de notas fiscais, as quais foram levantadas como mais necessárias a serem apresentadas dentro do sistema. As regras de apresentação das informações estão contidas nesta classe, a qual estará verificando o negócio, e efetuando as operações de consulta ao banco de dados a partir da classe implementada para interface com o banco. Abaixo apresenta-se o principal método da classe que é: busca_NotaFiscal, método sobrecarregado, pode ser chamado passando dois grupos de parâmetros, que seriam, o número da nota fiscal (cdNota) e o número da série da nota (dsSerie), ou passar um intervalo de datas (dtInicial e dtFinal) e o código do cliente, retornando um *DataTable* contendo todas as informações da capa da nota fiscal;

f) Item_Nota_Fiscal: classe que contém os atributos e métodos responsáveis pela apresentação das informações referentes aos itens da nota fiscal. Informações apenas

de leitura, não possuindo nenhum tipo de operação de banco de dados que não seja consulta. As informações da base são alimentadas a partir do sistema de sincronização buscadas no repositório central de dados. Abaixo apresenta-se o principal método da classe: busca_ItensNotaFiscal, depende basicamente como parâmetro, o número da nota fiscal (cdNota) e retornando um *DataTable* contendo todos os itens referentes a esta nota fiscal. Efetua a verificação das regras de negócio para apresentação do item e faz a busca na base de dados através da classe de interface com o banco de dados;

g) Duplicatas: os dados que esta classe trabalha são referentes aos documentos de cobrança emitidos ao cliente a partir do faturamento de uma nota fiscal. Os atributos e métodos constantes nesta classe foram descritos para apresentar estas informações de forma resumida ao usuário, onde ele poderá efetuar verificações referentes a pendências financeiras do cliente. As informações desta classe são provenientes do repositório central de dados e atualizada via sistema de sincronização. São informações de apenas leitura, não possuindo nenhum tipo de transação de atualização ou exclusão de registros a partir do sistema. O principal método desta classe é: busca_Duplicata, os parâmetros passados para este método são o código do cliente(cdCliente) e um intervalo de datas para pesquisa (dtInicial e dtFinal). O retorno desta classe é um *DataTable* com todos os dados que atenderam a solicitação do filtro. A classe irá verificar apenas as condições de negócio, por exemplo a verificação se a duplicata está realmente em aberto e se o cliente é válido. O acesso a dados será feito a partir da classe de interface implementada para acesso ao banco de dados;

h) Histórico_Cliente: esta classe trabalha com as informações referentes ao histórico financeiro do cliente com a empresa. Possui os atributos e métodos para operações sobre as informações de pagamentos, faturamentos mensais e anuais do cliente e ainda os limites de crédito. As informações acessadas por esta classe dentro da base de dados são apenas de leitura, não possuindo nenhuma transação de banco de dados que não seja leitura. As informações manipuladas por esta classe são provenientes do repositório central, que efetua todo o levantamento das informações e envia para o dispositivo os dados via sistema de sincronização de dados. O principal método implementado na classe: busca_HistoricoCliente, recebe como parâmetro o código do cliente (cdCliente) efetuando a busca no banco de dados via classe de interface com o banco e retornando um *DataTable* com todas as informações históricas do cliente;

i) Item: os atributos e métodos que constam nesta classe são necessários para a manipulação das informações de itens no sistema. Os itens são informações apenas para leitura e são atualizados no banco de dados a partir do repositório central via sistema de sincronização. Os métodos constantes nesta classe efetuam apenas operações de consulta ao banco de dados, não possuindo nenhum tipo de transação de atualização ou exclusão de dados. As informações são utilizadas basicamente por outras classes que dependem de informações cadastrais referentes a itens para poderem executar suas operações. Descreve-se o principal método: busca_Item, método responsável por trazer as informações cadastrais referente a item. Recebe como parâmetro o código do item (cdItem) e retorna para o sistema um *DataTable* com as informações do item solicitado. A busca dos dados no banco é feita via classe de interface com o banco de dados;

j) Estoque: a classe de estoque possui atributos e métodos para operação sobre as informações de estoque de itens. Estas informações são provenientes do repositório central, atualizada via sistema de sincronização e são apenas para leitura, não possuindo nenhuma transação de banco de dados que não seja consulta as informações. As informações geradas por esta classe são utilizadas basicamente pelo sistema em situações que seja necessário a avaliação do estoque de um item, como por exemplo, na consulta de itens que traz junto com ela o estoque do item. O método principal da classe é descrito: busca_Estoque, retorna as informações referente a quantidade em estoque de um determinado item. Recebe como parâmetro o código do item (cdItem) e retorna um *DataTable* com as informações de estoque. O acesso à base de dados para a busca das informações é feito via classe de interface de acesso ao banco de dados;

k) Planejamento_Fabricação: os atributos e métodos desta classe provém informações referentes aos planejamentos de fabricação de um item. Estas informações são importantes, pois dão ao usuário a facilidade de saber sobre todas as vendas e planejamentos de fabricação de um item, dando um horizonte ao representante quanto às datas de entrega de produtos e disponibilidade de estoques. As informações acessadas por esta classe são apenas leitura, não possuindo nenhuma transação de banco que efetue atualização ou exclusão de dados. Descreve-se o principal método da classe: busca_PlanejamentoFabricacao, recebendo como parâmetro o código do item (cdItem) este método retorna um *DataTable* com as informações de planejamento de

fabricação constantes para um item. Para acessar as informações, utiliza a classe de interface com o banco de dados para buscar os dados;

l) Venda_Efetuada: as informações de vendas efetuadas se referem a todas as transações de vendas existentes que envolvem o item informado em um determinado espaço de tempo. São vendas planejadas que geram necessidades de produção e vendas. Elas são utilizadas em previsões de vendas e planejamento de fornecimento de produtos, auxiliando no planejamento junto ao cliente de fechamento de grandes pedidos ou necessidades de planejamentos fechados de fornecimentos mensais. Os atributos e métodos desta classe disponibilizam as informações armazenadas na base de dados referente a estas transações, informações estas buscadas no repositório central e atualizando a base de dados local a partir do sistema de sincronização de dados. As informações são apenas de leitura, não existindo nenhum tipo e operação de atualização ou exclusão de dados. O principal método desta classe é descrito como: busca_VendaEfetuada, efetua a busca na base de dados de todas as vendas efetuadas para o referido item (cdItem), recebendo como parâmetro de entrada o código do item, e retornando um *DataTable* com as informações de venda. Acessa estas informações a partir da classe de interface com o banco de dados gerenciando apenas a parte das informações que deverão ser apresentadas;

m) Condição_Pagamento: possui métodos e atributos responsáveis pela apresentação das informações de condição de pagamentos do sistema. É uma classe apenas de leitura, buscando as informações do repositório central de dados via sistema de sincronização. A condição de pagamento é utilizada principalmente pela classe de Pedidos e Cliente, onde ambas possuem relacionamento direto com ela. Nos pedidos enviados ao repositório central, ela quem diz qual o número de parcelas que será efetuado o pagamento e o custo financeiro. O principal método da classe é descrito: busca_CondiçãoPagto, recebendo como parâmetro o código da condição de pagamento (cdCondicao) , ele efetua a busca no banco de dados a partir da classe de interface de acesso a dados e retorna para a solicitação um *DataTable* com as informações referentes à condição;

n) Condição_Entrega: possui métodos e atributos responsáveis pela apresentação das informações de condição de entregas dos pedidos. É uma classe apenas de leitura, buscando as informações do repositório central de dados via sistema de sincronização. A condição de entrega é utilizada principalmente pela classe de Pedidos e Cliente,

onde ambas possuem relacionamento direto com ela. Nos pedidos enviados ao repositório central, ela quem diz a taxa de cobrança de frete e o valor mínimo da cobrança caso não alcance a porcentagem mínima. O principal método da classe é descrito: busca_CondicaoEntrega, recebendo como parâmetro o código da condição de entrega (cdCondicaoEntrega) , ele efetua a busca no banco de dados a partir da classe de interface de acesso a dados e retorna para a solicitação um *DataTable* com as informações referentes a condição;

o) Transportadora: classe responsável por disponibilizar os dados referentes à transportadora no sistema, possui atributos e métodos para tal finalidade. As informações de transportadora são alimentadas a partir do repositório central, via sistema de sincronização. São informações apenas para leitura, não possuindo nenhum tipo de transação de banco para alteração ou exclusão de registros a partir do sistema. O principal método desta classe é explicado a seguir: busca_Transportadora, recebendo como parâmetro o código da transportadora (cdTransportadora) , ele efetua a busca no banco de dados a partir da classe de interface de acesso a dados e retorna para a solicitação um *DataTable* com as informações referentes a transportadora solicitada;

p) Representante: o sistema está ligado diretamente ao representante comercial usuário do sistema, onde é necessária uma classe para disponibilizar o acesso ao sistema a partir do representante informado. A classe é de apenas leitura, buscando seus dados a partir do repositório central de dados, não contendo nenhum tipo de transação de banco que não seja consulta. Os métodos e atributos da classe basicamente informam o nome (dsRepresentante) e código(cdRepresentante) do cliente e possui um campo de senha (dsSenha) para validação do acesso do usuário. Os principais métodos são descritos a seguir:

   - Busca_Representante: recebe como parâmetro o código do representante (cdRepresentante), e retorna um *DataTable* com as informações referentes ao código solicitado. Efetua a busca das informações no banco de dados a partir da classe de interface com o banco de dados implementados especificamente para este fim,
   - valida_SenhaRepresentante: este método efetua a verificação da senha do representante para permitir acesso ao sistema. Basicamente passa como

parâmetro a senha e o código do representante, e retorna um *boolean* dizendo se foi aceito ou não a autenticação do representante;

q) Cidade: a classe de cidade armazena basicamente a cidade, estado e um código de busca identificando-a no sistema. Classe apenas de leitura, está ligada basicamente ao cadastro de clientes, e utilizada também na classe de pedidos. Seus dados são carregados do repositório central a partir do sistema de sincronização. Possui atributos e métodos necessários para identificação de uma cidade explicado a seguir: busca_Cidade, método responsável por buscar e trazer as informações da cidade. Recebendo como parâmetro o código da cidade (cdCidade). O método retorna como resultado um *DataTable* com as informações pertinentes a esta cidade;

r) Banco_Dados: classe implementada especialmente para efetuar todos os tipos de transação de banco de dados, recebendo os parâmetros para consultas, inserções, atualizações e exclusões de dados, devolvendo à classe de negócio apenas o resultado necessário para a obtenção do resultado final da solicitação feita ao negócio. Em todos os retornos, ela tem como padrão retornar um *DataTable* com os dados solicitados, ou ainda com o grupo de dados modificado ou excluído. Abaixo, descreve-se brevemente os métodos constantes na classe:

   - carrega_Cliente: recebendo como parâmetro o código ou parte do nome do cliente, retornará os dados cadastrais do cliente. Método sobrecarregado,
   - carrega_ContatoCliente: recebe como parâmetro o código do cliente e parte do nome e retorna os dados do cadastro de contato do cliente. Método sobrecarregado,
   - carrega_Item: carrega as informações da tabela de itens, recebendo como parâmetro o código do item,
   - carrega_ItemPedido: recebe como parâmetro o código do pedido e retorna todos os itens constantes nas linhas do pedido,
   - carrega_ItemNotaFiscal: recebe como parâmetro o código da nota fiscal e a série e retorna todas as informações condizentes aos itens da nota fiscal,
   - carrega_PlanejamentoFabricacao retorna todas as informações de planejamento de fabricação do item. Recebe como parâmetro o código do item e retorna todas as datas planejadas de fabricação,

- carrega_VendaEfetuada: a tabela de vendas efetuadas possui as informações de todas as vendas de um item em andamento que ainda não foram atendidas por estarem em fabricação,
- carrega_CondicaoPagto: carrega todas as condições de pagamento da tabela, ou a condição desejada a partir do código informado. Método sobrecarregado,
- carrega_HistoricoFinanceiro: carrega o histórico financeiro do cliente, recebe como parâmetro o código do cliente e a empresa e retorna o histórico completo do cliente,
- carrega_Estoque: a partir de um item passado como parâmetro, retorna o estoque do item,
- carrega_CondicaoEntrega: carrega todas as condições de entrega da tabela, ou a condição desejada a partir do código informado. Método sobrecarregado,
- carrega_Duplicata: carrega as duplicatas a partir de um intervalo de datas e código de cliente, ou apenas pelo intervalo de data trazendo todas as duplicatas dos clientes .Método sobrecarregado,
- insere_ContatoCliente: método que recebe todas as informações de contato do cliente e efetua a inserção da informação na tabela,
- exclui_ContatoCliente recebendo o código do contato, efetua a exclusão do contato do cliente,
- atualiza_ContatoCliente: caso seja alterada alguma informação do contato, repassando todas as informações do contato para o método ele vão atualizar a informação na base de dados,
- insere_CapaPedido: recebe como parâmetros os dados da capa do pedido e grava na tabela correspondente,
- atualiza_CapaPedido: recebe como parâmetro os dados da capa do pedido, tanto os modificados como os não modificados e executa uma atualização no banco de dados,
- exclui_CapaPedido: efetua a exclusão do pedido, recebendo como parâmetro o código do pedido,
- insere_ItemPedido: efetua a inserção de um item no pedido, recebendo como parâmetros os dados de um item no pedido,

- atualiza_ItemPedido: ao se modificar as informações de um item no pedido, ele recebe como parâmetros todos os dados do pedido e efetua a atualização de dados na tabela,
- exclui_ItemPedido: recebe como parâmetro o item do pedido e efetua a exclusão do registro no banco,
- carrega_CodigoSequenciaPedido: ao ser chamado, verifica as tabelas de seqüência e carrega um novo código, efetuando a atualização da seqüência existente;

5.3 IMPLEMENTAÇÃO

Neste capítulo serão apresentados tópicos referentes a implantação do protótipo deste trabalho.

5.3.1 TÉCNICAS E FERRAMENTAS UTILIZADAS

Nos próximos tópicos, serão apresentadas algumas das principais ferramentas e técnicas utilizadas para o desenvolvimento deste trabalho. O principal destaque é dado para a forma de implementação em *Visual Basic .NET* e a base de dados *SQL Server 2000.*

5.3.1.1 Visual Studio .NET

É a mais nova versão de ferramentas para desenvolvimento de *softwares* da Microsoft orientada diretamente ao novo conceito de programação e arquitetura da plataforma .NET.

Com ela é possível apenas escrever códigos compatíveis com o *Software Developement Kit* (SDK) da *.NET Framework* que consiste em um composto de ferramentas e implementações independentes que facilitam os trabalhos do programador.

O *Visual Studio .NET* oferece editor de código fonte, editor de imagens e *forms*, compiladores, ajuda e outros dispositivos para facilitar o trabalho de desenvolvimento.

A figura 14 mostra a interface principal do aplicativo.

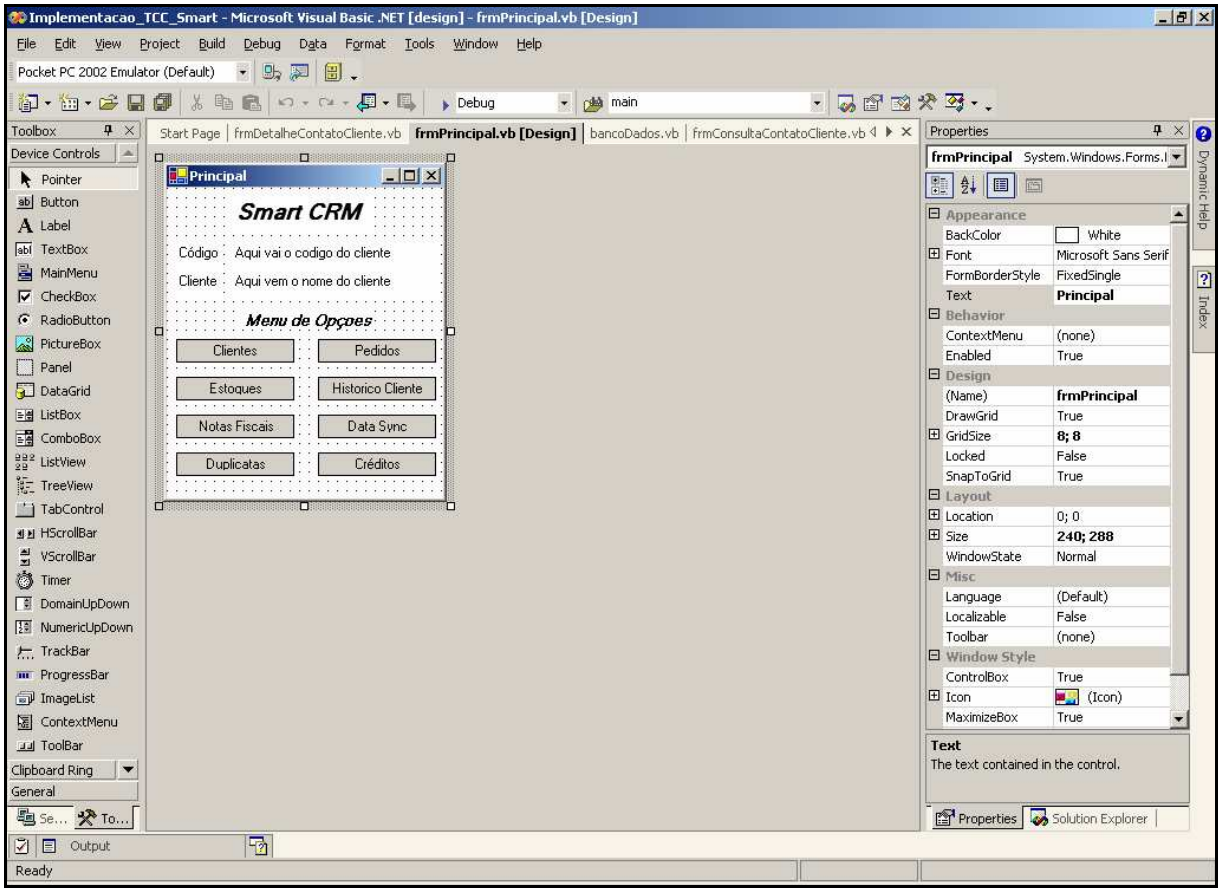


Figura 14: Interface do *Visual Studio .NET*

Ele suporta todas as linguagens hoje utilizadas no desenvolvimento para a plataforma como o *Visual Basic.NE*T, C#, C++, J#, e outras linguagens possíveis para desenvolvimento.

Sua *Integrated Development Environment* (IDE) é comum para todas as linguagens, diferente de outras versões do *Visual Studio* onde era necessário abrir a ferramenta de desenvolvimento para cada tipo de linguagem. Com o *Visual Studio .NET*, inicialmente abre-se a IDE do VS e depois simplesmente seleciona o tipo de projeto que estará sendo desenvolvido.

A figura 15 mostra a praticidade na escolha do tipo de linguagem e de projeto a ser trabalhada, dentro da mesma interface com usuário.

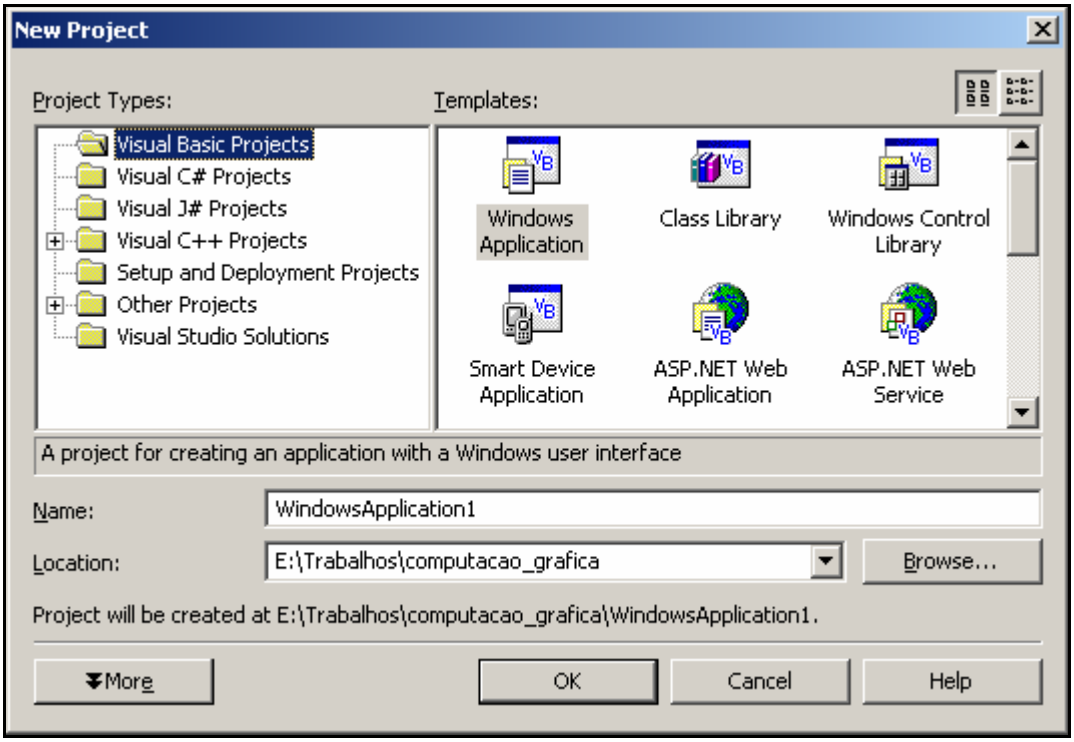


Figura 15: Tela de escolha de projeto do *Visual Studio .NET*

Outro grande diferencial comparando-se com as versões anteriores do *Visual Studio* é a disposição dos elementos da IDE: pois o sistema de menus e barras de ferramentas não variam, onde apenas alguns elementos das opções são desabilitados, porém os elementos de interface e de desenvolvimento são comuns para qualquer linguagem utilizada no desenvolvimento.

A edição do código fonte é efetuada em uma interface totalmente voltada ao auxílio do desenvolvedor, facilitando das mais variadas formas a programação, utilizando-se de agrupadores de procedimentos, âncoras explicativas de procedimentos e funções, autocompletar, paleta de objetos entre outros.

A figura 16 apresenta um breve exemplo dos auxílios do ambiente de desenvolvimento do *Visual Studio .NET*.

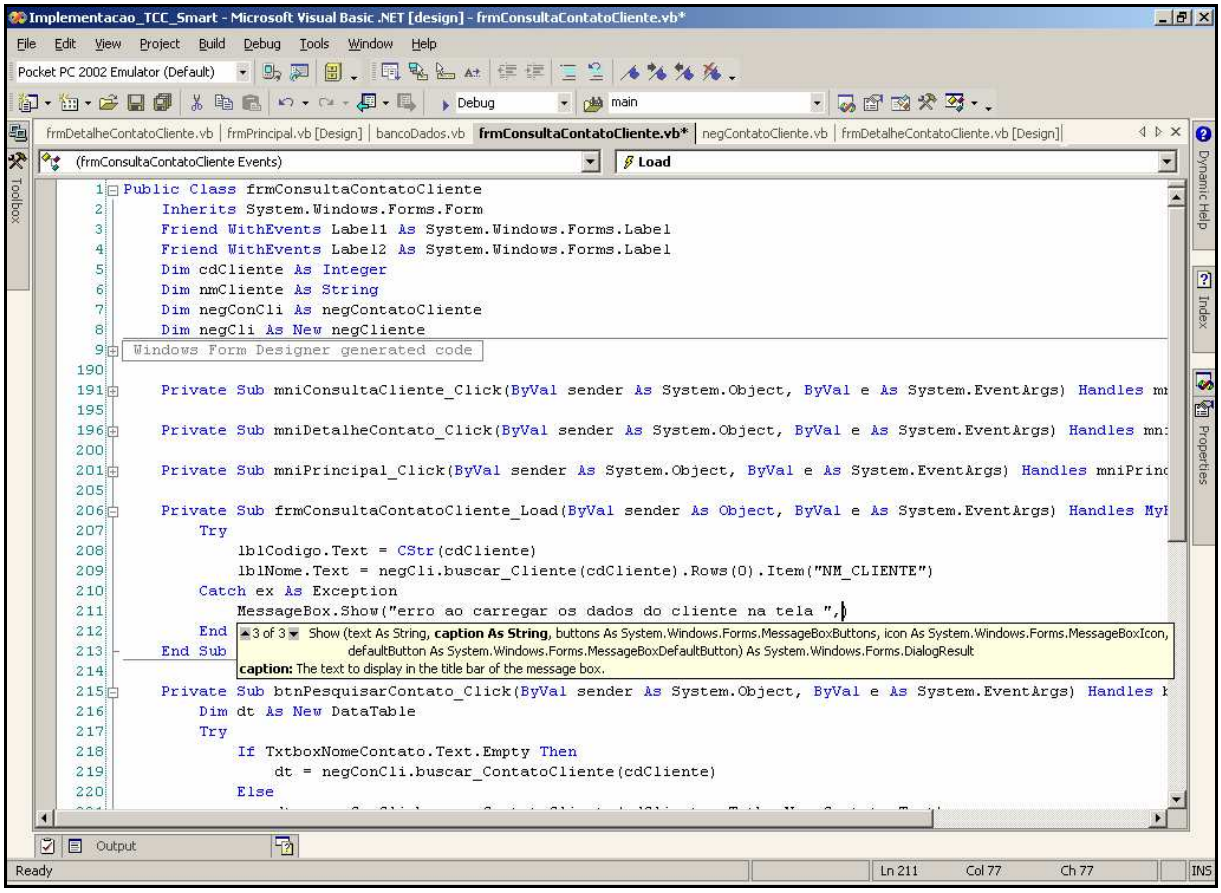


Figura 16: Ambiente de edição dos fontes do *Visual Studio .NET*

5.3.1.2 Banco de Dados SQL Server 2000

O *SQL Server* 2000 é o sistema de gerenciamento de banco de dados relacionais da *Microsoft*. Ele é resultado de mais de uma década de desenvolvimentos e estudos para chegar a um produto final e de grande sucesso.

Sua arquitetura é preparada de forma que possa garantir a estabilidade e escalabilidade, cumprindo trabalhos de missões críticas superando recordes digitais de armazenamento e gerenciamento de dados de sua categoria.

> O *SQL Server* 2000 não é apenas um aplicativo simples de banco de dados como o *Microsoft Access* ou um *Visual Fox Pro*, mas sim uma coleção de componentes e produtos que combinam holisticamente sistemas cliente/servidor, mas também sistema de *Data Storage*, recuperação de dados, e sistemas de análise e requerimentos para uma entidade ou organização. (Shapiro, 2001, p. 42)

Ele pode ser aplicado em casa para pequenos desenvolvimentos como também ser aplicado a grandes empresas ou companhias que possuem *Web Sites* com milhares de

transações por segundo ou apenas pequenas companhias com apenas algumas estações de trabalhos conectadas. Tudo isso personalizado para cada tipo de utilização.

Entre as principais funcionalidades e capacidades podem-se enumerar algumas, como o gerenciamento de transações de banco em SQL, otimizando códigos e solicitações ao banco de dados a partir de estruturas de *select*, por exemplo. Podem ser desenvolvidas *Stored Procedures* aproveitando os avanços da capacidade de programação dentro do ambiente *SQL Server* 2000. São previstas inúmeras funções disponíveis aos desenvolvedores.

O reforço na segurança e na integridade de dados deve ser ressaltado, pois toda a segurança e forma de transações foram revistas garantindo maior segurança aos aplicativos que acessam as bases de dados. Dentro do *SQL Server* 2000 tem-se ainda toda a segurança de acesso a dados a partir de configurações de *roles* de acesso aos usuários.

É prevista uma grande importância a sistemas de missão crítica, vinte e quatro horas e sete dias por semana e todas as manutenções do banco podem ser feita *online* sem necessidade de parar o banco para tais situações.

Existem várias versões do *SQL Server* 2000 conforme Shapiro (2001), onde se destaca para este trabalho a versão SQL 2000 Compact Edition.

5.3.1.3 Banco de dados SQL SERVER 2000 CE (MSDE)

A *Microsoft SQL Server 2000 Desktop Engine* (MSDE) é uma nova tecnologia que fornece armazenamento de dados local compatível com o *Microsoft SQL Server* 2000. Pode ser utilizado como uma solução de armazenamento de dados remota em sistemas *stand alone,* onde é necessário trabalhar desconectado com replicação de dados. Outra utilização é como um mecanismo de dados cliente/servidor alternativo ao mecanismo de banco de dados do Microsoft *Jet* do servidor de arquivos. Na versão *Compact Edition (CE)* suporta o ambiente *Windows* CE .NET 4.2.. Roda em aplicativos e dispositivos que possuem a plataforma *Windows* CE (*Windows* CE 4.2) como, por exemplo, os Pockets PC. Possui a mesma arquitetura e é compatível com todos as versões do *SQL Server* 2000, onde existem códigos e aplicativos que são facilmente portáveis para versões do sistema operacional *Windows* CE. Sendo assim é possível desenvolver aplicativos para plataforma *Windows* CE que possuam bases de dados relacionais garantindo integridade e qualidade na aplicação e nos dados.

Outra particularidade interessante e prática é o fato de se basear no mesmo mecanismo de dados do *SQL Server*, sendo assim a maioria dos projetos do *Microsoft Access* ou dos aplicativos cliente/servidor podem ser executadas sem nenhuma alteração em códigos fonte. Entretanto, diferente do *SQL Server* 7.0, o MSDE possui um limite de tamanho de banco de dados de dois *gigabytes*, não oferece suporte *Symmetrical Multiprocessing* (SMP) nos ambientes Windows 9x, e, quando usa replicação transacional, não pode ser um editor de replicação (embora possa agir como um assinante de replicação) e não possui nenhum ambiente de interface para administração do servidor.

5.3.2 IMPLEMENTAÇÃO DO PROTÓTIPO

A implementação do protótipo foi separada em camadas, onde cada uma delas representa um nível na hierarquia da aplicação. A primeira camada corresponde à camada de interface com o usuário, onde apenas as funções referentes à carga de dados de tela são executadas a implementadas no objeto de interface.

A figura 17 apresenta um trecho de código onde a partir de um evento de tela é solicitada uma informação a classe de negócio para que seja apresentada posteriormente.

As verificações necessárias dos objetos de tela para que se possa disparar a consulta também são feitas neste momento.

Fica claro na chamada da função, a qual passa os parâmetros de tela para a função de negócio e solicita o retorno de um objeto de *Data Table* que é uma estrutura que armazena informações de banco de dados. Este *Data Table* é utilizado como um *Data Source* pela grade da tela, que irá apresentar as informações retornadas do objeto de negócio.

```
266
267 Private Sub frmAcesso_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles MyBase.Load
268     Dim dr As SqlCeDataReader
269     TxtboxDataHora.Text = Date.Now.ToShortDateString & "-" & Date.Now.ToShortTimeString
270     Try
271         If conString Is Nothing Or conString.State = ConnectionState.Closed Then
272             conString = New SqlCeConnection("Data Source=\My Documents\POCKET.sdf")
273             conString.Open()
274         End If
275         dr = neg.carregar_Representante()
276         While dr.Read
277             CmbboxUsuario.Items.Add(dr.Item("NM_REPRESENTANTE"))
278         End While
279
280     Catch ex As Exception
281         MessageBox.Show("Erro ao carregar Combo de usuários " & ex.Message, "Atencao", MessageBoxButtons.OK,
282              MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1)
283     End Try
284 End Sub
```

Figura 17: Objeto de tela solicitando informações para objeto de negócio

A segunda camada refere-se à modelagem do negócio, onde as funções que dizem respeito à configuração na forma que os dados devem ser apresentados, e as regras para que eles possam aparecer, ficam centradas nesta etapa do fluxo de execução da aplicação.

Neste mesmo momento, que depois de verificada a solicitação do objeto de tela e das necessidades apresentadas, ele efetua o processamento da solicitação e fará a busca das informações a partir de uma interface com o banco de dados que representa a camada de acesso a dados da aplicação.

A chamada à classe de acesso a dados, que é implementada no modelo como uma interface, se dá pela instância do objeto na classe de negócio e a chamada do método referente à necessidade da informação.

Existe também a sobrecarga dos métodos em vários casos, onde dependendo do tipo de parâmetro passado, será efetuado o retorno dos dados.

Outro detalhe importante é de que no caso de solicitação de dados, todos eles são retornados encapsulados em um *Data Table*, que é um objeto de armazenamento de dados utilizado no *Visual Basic .NET* e que fornece várias operações sobre os dados que ela armazena.

Na figura 18 apresenta-se um exemplo de implementação na camada de negócio.

```
    Friend Function carregar_Representante() As SqlCeDataReader
        Dim dr As SqlCeDataReader
        Try
            dr = bd.carrega_Representante()

            If dr.Read Then
                Return dr
            Else
                Return Nothing
            End If

        Catch ex As Exception
            MessageBox.Show("Erro ao carregar cliente em NG " & ex.Message)
        End Try

    End Function
```

Figura 18: Exemplo de procedimento da classe de negócio

A terceira camada, que é a camada de acesso a dados, é implementada em uma interface específica apenas para manipulação de dados na base.

Todos os seus métodos têm algum tipo de interação direta com a base de dados efetuando funções de seleção de dados (*SELECT*), inserção de dados (*INSERT*), atualização de dados (*UPDATE*) e exclusão de dados (*DELETE*), montando estas funções e executando em banco a partir dos parâmetros enviados pelo objeto de negócio. Todos os métodos são funções (*FUNCTIONS*) que retornam um tipo *Data Table*, ou seja, buscam as informações na base de dados e encapsulam em um *Data Table* para retorná-lo para o objeto de negócio manipular.

A figura 19 apresenta um exemplo de método que efetua uma busca de dados na base e retorna um *Data Table* com as informações solicitadas a partir dos parâmetros repassados.

```
    'metodos especiais para cargas de dados normalmente sem parâmetros
    Friend Function carrega_Representante() As SqlCeDataReader
        Try
            Dim drComm As New SqlCeCommand("SELECT CD_REPRESENTANTE, NM_REPRESENTANTE, DS_SENHA " & _
                  " FROM REPRESENTANTE ", frmAcesso.conString)
            drComm.CommandType = CommandType.Text

            Dim dr As SqlCeDataReader
            dr = drComm.ExecuteReader(CommandBehavior.Default)

            Return dr
        Catch ex As SqlCeException
            MessageBox.Show("Erro ao carregar Representantes em BD " & ex.Message, "Atencao", MessageBoxButtons
                            MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1)
        Catch ex As Exception
            MessageBox.Show("Erro ao carregar Representantes em BD " & ex.Message, "Atencao", MessageBoxButtons
                MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1)
        End Try
    End Function
```

Figura 19: Exemplo de implementação de método da interface de acesso a dados

Outro ponto importante na implementação foi a forma da criação da base de dados, onde não se utiliza nenhum mecanismo de banco de dados para sua criação mas, é implementada manualmente a sua criação na primeira instância da aplicação, eliminando a possibilidade de erros em uma instalação mau sucedida.

Inicialmente é criada a base de dados sem nenhuma informação, a partir de comandos disponíveis na *Compact Framework .NET*, caso ela ainda não exista.

Depois disto, todos os comandos *Data Definition Language* (DDL) são executados um a um, repassados para o banco também a partir de métodos responsáveis pela execução destes.

A figura 20 ilustra uma parte do código onde é feita a verificação da existência da base de dados no dispositivo móvel.

```
124     Private Sub criaBD()
125         Try
126             'conferir se a base existe
127             If Not File.Exists("\My Documents\POCKET.sdf") Then
128                 'cria a base de dados e a conexao
129                 Dim baseEngine As New SqlCeEngine("Data Source=\My Documents\POCKET.sdf")
130                 baseEngine.CreateDatabase()
131                 conString = New SqlCeConnection("Data Source=\My Documents\POCKET.sdf")
132                 conString.Open()
133 
134                 'cria as tabelas (CLIENTE)
135                 Dim cmd As New SqlCeCommand(" CREATE TABLE CLIENTE ( " & _
136                  " CD_CLIENTE int PRIMARY KEY NOT NULL , " & _
137                   " CD_CONDICAO_ENTREGA int NOT NULL , " & _
138                     " CD_CONDICAO_PAGTO int NOT NULL , " & _
139                     " CD_REPRESENTANTE int NOT NULL , " & _
140                     " CD_TRANSPORTADORA int NOT NULL , " & _
141                     " NM_CLIENTE nvarchar (50)  , " & _
142                     " DS_CNPJ nvarchar (15) , " & _
143                     " DS_ENDERECO nvarchar (50) , " & _
144                     " FL_SITUACAO nvarchar (2) , " & _
145                     " DS_TELEFONE_COM nvarchar (15) , " & _
146                     " VL_MVC int , " & _
147                     " DS_BAIRRO nvarchar (50), " & _
148                     " DS_CIDADE nvarchar (50), " & _
149                     " DS_UF nvarchar (10)) ", conString)
150                 cmd.ExecuteNonQuery()
151                 'cria tabela de CONDICAO_ENTREGA
152                 cmd.CommandText = "CREATE TABLE CONDICAO_ENTREGA (CD_CONDICAO_ENTREGA int Primary Key NOT NULL
153                     " DS_CONDICAO_ENTREGA nvarchar (30)  , NM_LOCALIDADE_ENTREGA nvarchar (50) ," & _
154                     " VL_MINIMO_FRETE float ,PR_MINIMO_FRETE float) "
```

Figura 20: Trecho de código da criaçao da base de dados e suas tabelas

5.3.3 OPERACIONALIDADE DA IMPLEMENTAÇÃO

Basicamente, o protótipo conta com duas principais funções que são consultar dados e inserir dados.

A parte de consultas provém das informações coletadas a partir de um repositório central, sendo apenas leitura, e existe alguns módulos que fornecem a possibilidade de inserir dados na base do próprio dispositivo para um futuro envio a um repositório central de dados.

A seguir são exemplificadas situações onde são efetuadas consultas e inserções de dados.

5.3.3.1 Consultas básicas a informações somente leitura

A tela na figura 21 mostra um exemplo de consulta ao sistema.

Figura 21: Exemplo de Tela de Consulta

Basicamente, uma consulta fornecerá uma tela onde serão solicitados os filtros que poderão ser inseridos, e abaixo existirá uma grade trazendo os resultados da pesquisa, apresentando os principais campos da informação.

Clicando sobre a linha da grade, se tem a possibilidade de detalhar estas informações, onde o sistema levará para outra tela, conforme figura 22 mostrando todos os detalhes da informação obtida.

Figura 22: Exemplo de Detalhe de Informações

Na tela apresentada anteriormente (figura 22), na borda abaixo dos dados apresentados, pode também ser visualizado o menu que fornece a possibilidade de encaminhar o usuário para as telas anteriores ou retornar ao menu principal, facilitando a navegação pelo sistema.

Nenhum campo na tela pode ser editável, sendo todas as informações somente leitura. As consultas que são somente leitura, apresentando apenas informações recebidas são informações pertinentes a Clientes, Duplicatas, Programação de Produção, Estoques e Histórico de Clientes.

5.3.3.2 Manipulação e inserção de informações editáveis pelo usuário.

As informações que podem ser manipuladas e inseridas são aquelas que fornecem a possibilidade de inserção de novas informações, modificação das informações e gravação das modificações e exclusão de informações.

As telas de edição de dados também possuem a possibilidade de consultas na mesma forma que nas telas de consultas simples, porém nas telas que apresentam as informações detalhadas, existe um botão Gravar, o qual tanto grava uma nova informação no caso de ter sido ativada a função de inserção, ou grava uma modificação efetuada nas informações que estavam sendo verificadas.

A próxima imagem apresentada (figura 23) apresenta a tela de consultas de pedidos, onde, além da função de pesquisar, existe também a função de inclusão.

Figura 23: Exemplo de tela de Pedidos com Consulta de Inclusão

A figura 24 mostra a tela de pesquisa de contatos que conta também com a possibilidade de exclusão de informações.

Figura 24: Exemplo de tela com exclusão de dados

Conforme comentado, na próxima imagem (figura 25) a tela de detalhe de pedidos, onde as informações estão em *combos* que podem ser editados e salvos a partir do botão Gravar.

Figura 25: Exemplo de Tela com possibilidade de edição de dados

As informações que podem ser editadas, inseridas e excluídas são basicamente as informações referentes a pedidos, desde que este pedido não tenha sido emitido e enviado ao repositório central.

5.4 RESULTADOS E DISCUSSÃO

O protótipo rodando sobre a plataforma .NET implantado em um dispositivo móvel apresentou os resultados esperados. A performance não se mostrou muito boa, porém o

principal motivo se dá pelo fato de estar rodando em um emulador, e não diretamente em um aparelho *Pocket PC*.

A plataforma promete que o desenvolvimento do sistema não influencia o tipo de dispositivo e ambiente que será implantada a aplicação, porém foi verificado que existem diferentes paradigmas para desenvolvimento em uma plataforma cliente/servidor (*Windows*) e uma plataforma para dispositivos móveis (*Windows CE*). Ao desenvolver para uma plataforma *Pocket PC*, os recursos encontrados no VB .NET tradicional não estão todos disponíveis para as bibliotecas específicas para o *Windows .NET CE* , sendo necessário uma verificação mais apurada das funcionalidades disponíveis para desenvolvimento.

Outro detalhe é o consumo de recursos do equipamento, pois um sistema de grande porte como o apresentado no trabalho exposto, depois de carregado, praticamente utiliza 50% dos recursos disponíveis em um equipamento de configurações já avançadas.

A criação e manutenção da base de dados também são mais complicadas, pois não se consegue criar a base a partir de *scripts* ou ferramentas automatizadas, sendo necessário implantar na própria implementação funções de criação das bases de dados, onde em sistemas com um modelo de dados mais elaborado gera problemas e onera todo o código. O lado positivo deste fato é que não existe a preocupação com instalações anteriores no dispositivo, bastando apenas rodar a aplicação em um primeiro momento e ela cuida de configurar o acesso e criação de base de dados.

A interface possui um bom nível de configuração, onde pode-se desenvolver interfaces que facilitam a vida do usuário durante os trabalhos realizados. O acesso à internet para a atualização ficou comprometido, pois a maioria dos aparelhos não dispõe de comunicação que não seja com outro computador *desktop.*

6 CONCLUSÕES

O CRM se mostra hoje como o principal ingrediente na fórmula do sucesso empresarial no mercado. Com ele é possível apresentar um diferencial tanto aos seus clientes quanto aos seus futuros clientes, tomando ações sobre informações colhidas junto ao cliente e encantando o cliente de tal forma a levá-lo a uma parceria e uma fidelidade lucrativa para ambas as partes.

Com a qualidade dos produtos apresentada hoje em dia, sendo que sem qualidade mais ninguém se mantém no mercado, qualidade deixou de ser um diferencial marcante, instruindo as empresas a buscarem um novo conceito de diferencial para buscar a fidelidade do cliente.

É neste momento que o CRM mostra toda sua importância, pois uma estratégia bem formulada com certeza levará a empresa ao topo do *ranking* da competição mercadológica, ou seja, a tornará a concorrente de maior peso do mercado.

Mas para se conseguir todo este diferencial e implantar uma estratégia vencedora, as ferramentas que auxiliarão neste processo têm que estar alinhadas às necessidades dos usuários.

E como CRM basicamente é informação, a primeira necessidade que aparece é a coleta destas informações em campo e a disponibilidade delas da melhor forma possível com a maior praticidade imaginável. É com este conceito que se buscou o desenvolvimento deste trabalho de um sistema de CRM operacional que rodasse em uma plataforma *Pocket PC* e *Windows .NET CE*.

Desenvolvendo o sistema, onde ele apresenta todas as movimentações do cliente com a empresa, as posições de situação de produto e todas as informações comerciais do cliente, fica bem mais fácil para um vendedor ou representante acompanhar e tomar decisões estratégicas sobre um determinado cliente, sendo isto feito da forma mais ágil e prática possível.

E a plataforma .NET se mostrou bem preparada para atender esta nova necessidade do mercado, trazendo uma ferramenta flexível e produtiva para desenvolvimento de aplicações para a gama de dispositivos móveis fornecidos hoje no mercado.

A facilidade de desenvolvimento fica bem próxima da facilidade para se desenvolver para o ambiente cliente/servidor com ferramentas similares, atendendo a contento o desenvolvimento para aplicativos *Smart Devices*.

Os objetivos foram todos alcançados durante o desenvolvimento do trabalho, onde inicialmente foi necessária uma readaptação para o desenvolvimento focado em dispositivos móveis e um maior levantamento de informações, pois as documentações para este ambiente e trabalhos correlatos para este ambiente são encontrados em pequena escala.

O fato de ser uma tecnologia nova e ser proprietária, não se apresenta muitos estudos de entidades livres prendendo o desenvolvedor apenas à documentação técnica disponibilizada pela própria Microsoft.

A flexibilidade de desenvolvimento da interface com o usuário foi feita de forma prática e a separação do desenvolvimento em camadas se mostrou uma solução bem aceita pela linguagem que trabalha muito bem com a parte de orientação a objetos. As classes de apoio disponibilizadas pela *Framework .NET* realmente aumentam a produtividade, dando um nível de abstração para o desenvolvedor dificilmente verificado em qualquer outra linguagem.

O problema ainda é o acesso à internet e a atualização de dados, pois poucos dispositivos oferecem conexões sem fio e que por sua vez também oferecem outros poucos tipos de conectividade o que impossibilita trabalhar conectado, ou até mesmo dificulta a atualização via *web services*.

Verificada a tendência da tecnologia que tem se desenvolvido a passos largos, com certeza em um curto espaço de tempo proverá infraestrutura para desenharmos este ambiente ideal que suporte as novas aplicações que estão por surgir para dispositivos móveis.

Hoje em dia é impossível pensar em desenvolver uma nova solução que não seja compatível com dispositivos móveis, estes que por sua vez fornecem muito mais praticidade para um mundo moderno que se vive hoje.

6.1 EXTENSÕES

Este trabalho ainda pode ser explorado, buscando novas tendências que venham a surgir no decorrer dos anos referentes à CRM, pois esta filosofia corporativa vive em

constante mutação sempre buscando inovar para atender e encantar o cliente da melhor forma possível.

Na implementação podem ser melhorados os meios de atualização de dados e ainda disponibilizar a possibilidade de estar se trabalhando conectado com o dispositivo, onde não se teria mais uma situação *stand alone*, mas sim teria um acesso direto à base de dados a partir do dispositivo, não tendo mais os problemas de se trabalhar com informações desatualizadas.

Verificar na implementação uma melhor forma de gerenciamento de memória e consumo com a interface com usuário, pois como este tipo de implementação exige uma interface rica em detalhes, acaba onerando o dispositivo.

Por último, seria aproveitar toda a análise do sistema e implementá-lo em Java e efetuar uma comparação e atribuir os pontos positivos e negativos de desenvolvimento de uma aplicação nas duas plataformas.

REFERÊNCIAS BIBLIOGRÁFICAS

BLANCO, Luis Miguel. **Programación en Visual Basic.NET.** Madrid: Grupo Eidos, 2002. 725 p.

BRETZKE, Mirian. **Marketing de relacionamento e competição em tempo real**: com CRM (*Customer Relationship Management*). São Paulo: Atlas 2000, 224 p.

CAUTELA, Alcinei Lourenço; POLLONI, Enrico Giulio Franco. **Sistemas de informação na administração de empresas.** São Paulo: Atlas 1978, 175 p.

CHAMPION, Michael; FERRIS, Cris;NEWCOMER, Eric; ORCHARD, David. **Web service Architecture.** USA, [2002]. Disponível em: <http://www.w3.org/TR/2002/WD-ws-arch-20021114/ >. Acesso em 21 out. 2004.

CUNHA, Davi. **Web services, soap e aplicações web**. São Paulo, [2003]. Disponível em: <http://devegne.netscape.com/community/WebServicesSOAPeAplicaçõesWeb.htm.>. Acesso em: 8 fev. 2004.

DELANEY, Kalen ***Inside*** **Microsoft SQL Server 2000.** Washington: Microsoft Press: 2001.

DEITEL, H. M. **Visual Basic.NET:** como programar**.** 2. ed. Tradução Célia Yumi Okano Taniwaki. São Paulo: Pearson Education do Brasil, 2004. 1088 p.

DEPINÉ, Fábio Marcelo**. Protótipo de software para dispositivos móveis utilizando JavaME para cálculo de regularidade em rally**. 2002. 55 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais,Universidade Regional de Blumenau, Blumenau.

GREENBERG, Paul. **CRM na velocidade da luz:** conquista e lealdade de clientes em tempo real na internet. 1. ed. Tradução Reinaldo Marcondes. Rio de Janeiro: Campus, 2001. 409 p.

HADDAD, Renato. **Visão Geral do .NET *Compact Framework***, São Paulo, [2004]. Disponível em: <http://www.microsoft.com/brasil/msdn/Tecnologias/netframework/Default.mspx>. Acesso em: 10 ago. 2004.

LAUDON, Kenneth C.; LAUDON, Jane Price. **Sistemas de informação**. Tradução Dalton Conde de Alencar. Rio de Janeiro: LTC, 1999.

MSDN MICROSOFT. **Microsoft .NET, *framework* e aplicativos WEB.** 1. ed. Tradução Izabel Cristina de Mendonça Santos. Rio de Janeiro: Campus, 2001. 177 p.

ROMAN, Steven. **Linguagem VB.NET, o guia essencial:** um livro de referência**.** 1. ed. Tradução Kátia Roque. Rio de Janeiro: Campus, 2002. 697 p.

SCHAEFER, Carina. **Protótipo de aplicativo para transmissão de dados a partir de dispositivos móveis aplicado a uma empresa de transporte**. 2004. 53f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

SCRIBNER, Kenn. ***Appied SOAP:*** *implementing .NET XML web services.* 1. ed. EUA: SAMS, 2002. 415 p.

SHAPIRO, Jeffrey R. ***SQL Server 2000:*** *the complete reference*. 1. ed. EUA: McGrall-Hill/Ousborne, 2001. 961 p.

SWIFT, Ronald. **CRM, *customer relationship management*:** o revolucionário marketing de relacionamento com clientes. 1. ed. Tradução de Flávio Deny Steffen. Rio de Janeiro: Campus, 2001. 493 p.

WIGLEY, Andy; WHEELWRIGHT, Stephen. ***Microsoft .NET compact framework****: core reference.* Washington: Microsoft Press, 2003.