UNIVERSIDADE DE FEDERAL DE VIÇOSA
CAMPUS UFV - RIO PARANAÍBA

LEANDRO CAIXETA FERREIRA

DESENVOLVIMENTO DE UM SISTEMA PARA DISPOSITIVOS MÓVEIS PARA AUTOMATIZAÇÃO DE OPERAÇÕES BARTER

RIO PARANAÍBA – MINAS GERAIS
2013

LEANDRO CAIXETA FERREIRA

DESENVOLVIMENTO DE UM SISTEMA PARA DISPOSITIVOS MÓVEIS PARA AUTOMATIZAÇÃO DE OPERAÇÕES BARTER

Monografia, apresentada ao Curso de Sistemas de Informação da Universidade Federal de Viçosa - Campus UFV - Rio Paranaíba como requisito para obtenção do título de bacharel em Sistemas de Informação.

Orientador: Thiago Pirola Ribeiro

RIO PARANAÍBA – MINAS GERAIS
2013

LISTA DE FIGURAS

FIGURA 1 - ESPECIFICAÇÕES TÉCNICAS DO NOVO IPAD® (APPLE, 2012E)..... 4
FIGURA 2 - CAMADAS DO IOS™ (APPLE, 2012A)..... 6
FIGURA 3 - UMA JANELA DE PROJETO NO XCODE (APPLE, 2012F)..... 7
FIGURA 4 – INSTRUMENTS (APPLE, 2012F)..... 8
FIGURA 5 - MODELO MVC (MARZULLO, 2012)..... 11
FIGURA 6 - MODELO MVC NO IOS™ (MARZULLO, 2012)..... 12
FIGURA 7 - PESQUISA DE CLIENTES (OLIVEIRA E COSTA, 2011)..... 15
FIGURA 8 - FLUXOGRAMA DAS ETAPAS..... 18
FIGURA 9 - DIAGRAMA DE CASO DE USO..... 24
FIGURA 10 - DIAGRAMA DE CLASSES..... 25
FIGURA 11 - DIAGRAMA ENTIDADE-RELACIONAMENTO..... 26
FIGURA 12 - TRECHO DE CÓDIGO DA CLASSE COTACAODB..... 27
FIGURA 13 - TELA INICIAL..... 30
FIGURA 14 - CONSULTA DE CLIENTES..... 31
FIGURA 15 - TELA DE SIMULAÇÕES..... 32
FIGURA 16 - NOVA SIMULAÇÃO – INFORMAR DADOS..... 33
FIGURA 17 - NOVA SIMULAÇÃO - SELECIONAR PRODUTOS..... 34

LISTA DE ABREVIATURAS E SIGLAS

API *Application Programming Interface* (Interface de Programação de Aplicativos)

ARC *Automatic Reference Counting* (Contador Automático de Referência)

BI *Business Intelligence* (Inteligência Empresarial)

CPR Cédula do Produtor Rural

CRM *Customer Relationship Management* (Gestão de Relacionamento com o Cliente)

ERP *Enterprise Resource Planning* (Sistema Integrado de Gestão Empresarial)

IDE *Integrated Development Environment (*Ambiente Integrado de Desenvolvimento)

IHC Interação Humano-Computador

MVC *Model-View-Controller* (Modelo-Visão-Controle)

PIB Produto Internxo Bruto

UML *Unified Modeling Language* (Linguagem Unificada de Modelagem)

XML *eXtensible Markup Language* (Linguagem de Marcação Extensiva)

SUMÁRIO

1 INTRODUÇÃO 1

2 REFERENCIAL TEÓRICO 4

2.1 iPad® 4

2.2 IOS™ 4

2.2.1 CARACTERÍSTICAS 5

2.2.2 GERENCIAMENTO DE MEMÓRIA 5

2.2.3 ARQUITETURA 6

2.2.4 XCODE E LICENCIAMENTO 7

2.3 OBJECTIVE-C 8

2.4 PERSISTÊNCIA DE DADOS 9

2.5 ARQUITETURA MVC (*Model-View-Controller*) 10

2.6 INTERAÇÃO HUMANO-COMPUTADOR 12

3 TRABALHOS RELACIONADOS 14

4 METODOLOGIA 18

4.1 ETAPAS 18

4.2 RECURSOS 20

5 DESENVOLVIMENTO 21

5.1 ANÁLISE DO SISTEMA ATUAL 21

5.2 LEVANTAMENTO DE REQUISITOS 21

5.2.1 REQUISITOS FUNCIONAIS 22

5.2.2 REQUISITOS NÃO FUNCIONAIS 22

5.3 MODELAGEM UML 23

5.3.1 DIAGRAMA DE CASO DE USO 23

5.3.2 DIAGRAMA DE CLASSES 24

5.4 MODELAGEM DO BANCO DE DADOS 25
5.5 CONSTRUÇÃO DO BANCO DE DADOS 26
5.6 IMPLEMENTAÇÃO 28
5.7 TESTES 28
5.8 APRESENTAÇÃO DO SOFTWARE 29
6 RESULTADOS OBTIDOS 35
7 CONCLUSÃO 36
8 TRABALHOS FUTUROS 37
9 BIBLIOGRAFIA 38

1 INTRODUÇÃO

O estado de Minas Gerais dispõe de um vasto território, grandes reservas de água e solo fértil. Esses fatores contribuem para que o estado ganhe destaque no cenário nacional no setor de agronegócios. Minas Gerais representa 12,4% do PIB (Produto Interno Bruto) nacional do setor segundo GOVERNO DE MINAS GERAIS (2012). Segundo TAGUCHI (2010), um tipo de operação básica tem movimentado milhões de reais nas transações de agronegócios no Brasil. Esse tipo de operação é uma tendência financeira sustentável para produtores rurais e fornecedores de insumos agrícolas. O antigo sistema de trocas, escambo ou permuta que remetem ao início do relacionamento mercantil entre índios e colonizadores europeus agora é conhecido como Operações Barter. Esse tipo de operação está facilitando os negócios de muitos produtores rurais, que nem sempre encontram facilidade em obter crédito no mercado para aquisição de insumos (sementes, fertilizantes e defensivos) a serem utilizados na produção. Em contrapartida, também se torna interessante para os fornecedores desses produtos, que encontram nas Operações Barter uma maneira de comercializar seus produtos, diminuindo o risco de inadimplência.

A Operação Barter consiste em antecipar insumos agrícolas para os produtores rurais, que emitem uma CPR (Cédula do Produtor Rural) comprometendo-se entregar um determinado volume de produção, proporcional aos valores dos produtos adquiridos.

Atualmente, as Operações Barter realizadas em uma determinada empresa da região, são realizadas através de um software utilizado em notebooks e desktops convencionais. A administração do sistema é de responsabilidade do Departamento de Operações Comerciais, que é responsável pela atualização dos dados e aprovação das negociações. O sistema atual não possui uma interface exclusiva para o administrador e essa ausência dificulta a manutenção dos dados e a comunicação entre os envolvidos, além de prejudicar o processo da operação como um todo. Na maioria das vezes, as Operações Barter são realizadas nas fazendas dos produtores que nem sempre dispõe de um ambiente confortável para utilização dos dispositivos. Além disso, o controle das negociações é realizado em planilhas, dificultando uma análise rápida das informações.

Este projeto tem como objetivo o desenvolvimento de um sistema para realizar as Operações Barter da empresa ABC. O sistema será desenvolvido para a plataforma iOS™ e será utilizado, inicialmente, no *tablet* iPad®. Futuramente esse sistema poderá ser também

utilizado no iPhone®. O sistema visa obter uma maior agilidade e mobilidade nas Operações Barter facilitando o processo através de uma interface simples e intuitiva.

A necessidade da análise mais profunda das informações que envolvem as negociações é primordial para o Departamento de Operações Comerciais. Através da análise, os tomadores de decisão da empresa podem direcionar os negócios e elaborar estratégias a fim de aumentar os lucros e estimular as Operações Barter.

Sem uma base de dados centralizada e bem estruturada torna-se difícil a criação de relatórios, estimativas e gráficos. Buscando uma maneira de centralizar as Operações Barter da empresa ABC, o sistema deve possibilitar uma futura integração com outros sistemas já utilizados na empresa. A base de dados deve ser planejada para uma futura integração com os sistemas de ERP (*Enterprise Resource Planning*) e BI (*Enterprise Resource Planning*) da empresa. O sistema de ERP futuramente funcionará como um servidor, onde será possível gerenciar as informações referentes às Operações Barter. Além disso, o ERP irá disponibilizar as informações que serão lidas pelo dispositivo móvel e receberá as informações das negociações realizadas no sistema móvel.

A Apple® é uma empresa já consolidada no mercado de dispositivos móveis e fornecedor do Sistema Operacional iOS™. Esse sistema operacional dispõe de vários recursos sofisticados e poderosos, interface elegante e simples, além da segurança aplicada ao sistema. O iOS™ dispõe de um grande portfólio de aplicativos, que podem ser utilizados para as mais diversas necessidades corporativas. Dentre os diversos produtos da Apple®, destaca-se o *tablet* denominado iPad®, sendo o mais vendido no mundo. Segundo Pontual (2012) o iPad® domina o mercado desse segmento, com 62% dos *tablets* vendidos em 2012. A concorrência do mercado global está cada vez mais acirrada, o que impulsiona o investimento tecnológico das empresas em novas tecnologias, buscando uma maneira de se sobressaírem no mercado. Segundo a IDC Brasil (2012), foram vendidos 370 mil tablets no primeiro trimestre de 2012, destes, 12% foram para o mercado corporativo. O iPad® possui um *designer* elegante e dimensões de tela consideradas adequadas para trabalhar com Operações Barter.

Além das qualidades e indicadores citados anteriormente, foram considerados diversos fatores para a escolha do hardware e software a serem utilizados no desenvolvimento do sistema, sendo os principais: Mobilidade, Segurança, Usabilidade e Robustez. Baseado nessas análises, a empresa ABC optou por desenvolver o sistema para a plataforma iOS™ que será executado no *tablet* iPad®. Com o uso desses dispositivos, os vendedores terão maior

facilidade em realizar as negociações, que em sua grande maioria são realizadas nas fazendas dos clientes, onde a mobilidade e usabilidade tornam-se aspectos imprescindíveis.

2 REFERENCIAL TEÓRICO

Nessa seção serão abordados os principais temas relacionados ao desenvolvimento de aplicativos para dispositivos móveis da Apple®.

2.1 IPAD®

A Figura 1 ilustra as especificações técnicas referentes às dimensões e botões do dispositivo iPad® modelo 3ª geração, dispositivo onde será realizado os testes da aplicação.

Figura 1 - Especificações Técnicas do Novo iPad® (Apple, 2012e)

2.2 IOS™

O iOS™ é o sistema operacional desenvolvido pela Apple® e utilizado nos dispositivos móveis por ela fabricados. O sistema foi lançado em janeiro de 2007, com o nome de iPhone OS. Inicialmente, o sistema operacional foi desenvolvido apenas para o iPhone® (telefone celular fabricado pela Apple®), posteriormente passou a ser utilizado em todos os dispositivos móveis que a Apple® lançou a partir de então; como iPod Touch(tocador de mídia portátil), iPad® (*tablet*) e Apple TV®(central multimídia).

2.2.1 CARACTERÍSTICAS

O iOS™ é um sistema operacional voltado exclusivamente para dispositivos móveis, nos quais os recursos como memória e processador são limitados se comparados a computadores convencionais. Com isso, ao desenvolver as aplicações, é necessário levar em consideração essas limitações, que por sua vez, possuem características particulares.

Cabe ao desenvolvedor conhecer e definir quais dispositivos darão suporte a aplicação. É possível bloquear a execução da aplicação em dispositivos mais antigos, que possuem um poder computacional inferior aos mais atuais. O tempo de reposta é uma questão que deve ser levada em consideração. Caso a aplicação demore mais de cinco segundos para executar em algumas situações, o iOS™ pode considerá-la em estado de ausência de resposta e suspender sua execução. A Apple® adotou essa medida de segurança para evitar que aplicações possam travar todo o sistema. O tamanho da memória e a resolução da tela são outros quesitos importantes que requerem certo cuidado, devido às variações existentes entre os dispositivos.

2.2.2 GERENCIAMENTO DE MEMÓRIA

Até a versão 4 do iOS™, era necessário ter muita atenção em relação ao gerenciamento da memória. Essa tarefa era de responsabilidade do desenvolvedor, que codificava comandos para liberar os objetos quando estes não fossem mais utilizados. Essa prática deveria ser utilizada para poupar recursos do sistema, entretanto sua má utilização poderia resultar em sobrecarga e travamento da aplicação. A partir da versão 5, o iOS™ incorporou um recurso bastante útil chamado de ARC (*Automatic Reference Counting*), responsável por fazer a liberação da memória automaticamente quando necessário. Esse recurso é opcional, podendo o desenvolvedor optar por continuar gerenciando manualmente a memória, como era feito em versões anteriores.

2.2.3 ARQUITETURA

A arquitetura do iOS™ é constituída de quatro camadas principais: *Cocoa Touch, Media, Core Services e Core OS*, conforme ilustrado na Figura 2 e descritas a seguir:

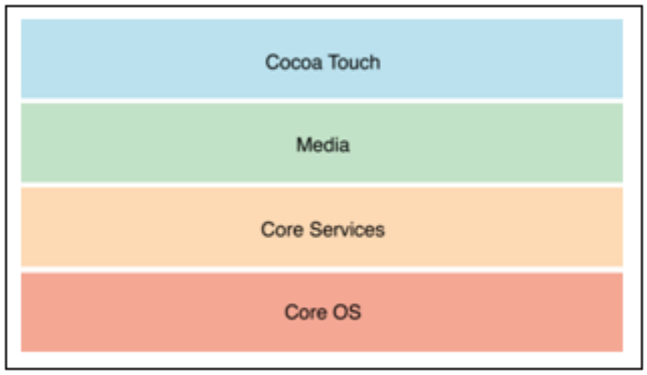

Figura 2 - Camadas do iOS™ (Apple, 2012a)

- *Cocoa Touch*: Representando a camada de mais alto nível na arquitetura do iOS™, é nessa camada que o desenvolvedor interage para construir suas aplicações. O *Cocoa Touch* é uma API (*Application Programming Interface*) desenvolvida pela Apple® especificamente para o sistema operacional iOS™. Essa API foi construída sobre a linguagem Objective-C e permite ao desenvolvedor o controle de alguns recursos como multitoque, interface gráfica, comunicação com arquivos, entre outros recursos.
- *Media*: Responsável por gerenciar animações, áudio, vídeo e algumas tecnologias utilizadas para o desenvolvimento de jogos.
- *Core Services*: Camada que disponibiliza alguns dos principais serviços para o desenvolvedor, como o acesso a banco de dados e manipulação de arquivos.
- *Core OS*: Camada responsável pelo gerenciamento de energia, sockets, certificados, dentre outros recursos importantes. Essa camada é considerada o núcleo do iOS™. Também é responsável por gerenciar a parte da segurança e comunicação do sistema com o mundo externo.

2.2.4 XCODE E LICENCIAMENTO

O desenvolvimento das aplicações para o iOS™ é realizado através do Xcode. O Xcode é uma IDE (*Integrated Development Environment*) livre disponibilizada pela Apple®, que pode ser encontrada na *Apple Store* (Apple, 2012d). A Figura 3 ilustra a interface do Xcode com um projeto denominado *HelloWord* aberto.

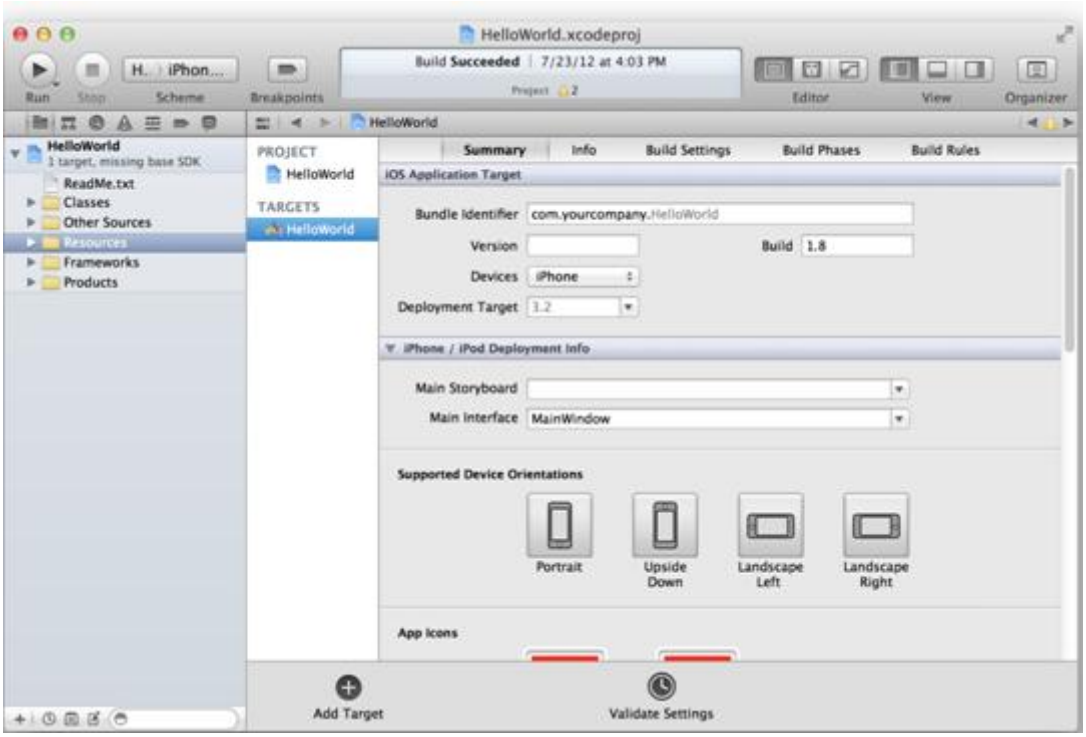

Figura 3 - Uma Janela de Projeto no Xcode (Apple, 2012f)

Apesar da ferramenta de desenvolvimento ser gratuita, para testar as aplicações desenvolvidas em dispositivos reais ou publicá-los na *Apple Store* é necessário ser um desenvolvedor licenciado pela Apple®. Caso contrário, as aplicações desenvolvidas só serão executadas no iOS Simulator - ferramenta disponibilizada juntamente com o Xcode para simular a execução da aplicação em dispositivos reais.

Existem dois tipos de licenças comerciais para desenvolvedores, a *Standard* e a *Enterprise*. A licença *Standard* é individual e custa atualmente 99 dólares anuais, sendo possível testar as aplicações em dispositivos reais e também publicá-los na *Apple Store*. A licença *Enterprise*, por sua vez, custa 299 dólares e possui os mesmos benefícios da licença *Standard*, com o diferencial de ser permitido habilitar mais de um desenvolvedor na mesma licença.

O Xcode disponibiliza uma ferramenta que permite analisar o desempenho dos aplicativos iOS™ durante a execução no simulador ou em um dispositivo real. O ambiente denominado *Instruments* coleta dados do aplicativo em execução e apresenta graficamente. Esta representação gráfica das informações é conhecida como Linha do Tempo. Podem ser coletadas informações sobre o uso da memória, atividade do disco e atividade de rede. A Linha do Tempo pode exibir todos os tipos de informações lado a lado, permitindo correlacionar o comportamento geral de sua aplicação, não apenas o comportamento em uma área específica. A Figura 4 ilustra o ambiente *Instruments*.

Figura 4 – Instruments (Apple, 2012f)

2.3 OBJECTIVE-C

O Objective-C é uma linguagem de programação que foi criada por Brad Cox e Tom Love na década de 80. A linguagem foi criada para suprir necessidades que os autores encontravam ao desenvolver seus projetos utilizando a linguagem C pura. Entre as dificuldades encontradas por eles, destacavam-se a dificuldade para reutilização de código e a troca de mensagens entre determinadas estruturas (MILANI, 2012). Em 1988, a empresa NeXT™, cujo proprietário era Steve Jobs, adquiriu e licenciou o Objective-C aumentando a popularidade da linguagem no mundo inteiro. Anos depois, em 1996, a Apple® adquiriu a empresa NeXT™ e acertou o retorno de Jobs à empresa. Atualmente, o Objective-C é a

principal linguagem de programação utilizada para o desenvolvimento de aplicações para o ambiente iOS™, sistema operacional utilizado nos dispositivos móveis da Apple.

O Objective-C incorpora conceitos de orientação a objetos à linguagem C padrão, além oferecer suporte a troca de mensagens. Qualquer código fonte desenvolvido em C pode ser compilado em um compilador Objective-C. Essa linguagem, entre outras palavras, é uma extensão da linguagem C combinada com a *Smalltalk*, umas das primeiras linguagens de programação orientadas a objetos.

2.4 PERSISTÊNCIA DE DADOS

Na computação, o termo persistência de dados se refere ao armazenamento não volátil de dados, ou seja, mesmo após a aplicação ser finalizada, os dados permanecem armazenados em algum dispositivo de armazenamento como discos rígidos e cartões de memória. Grande parte das aplicações construídas para o iOS™ realiza algum tipo de armazenamento de dados durante sua execução (MILANI, 2012). O armazenamento dos dados pode estar presente em aplicações simples que armazenam, por exemplo, configurações preferenciais dos usuários ou em aplicações complexas, que armazenam grande volume de dados. Para persistir dados no iOS™, atualmente três mecanismos são mais utilizados: arquivos, o banco de dados SQLite™ e o *framework Core Data*. Segundo MARZULLO (2012) e MILANI (2012):

- Arquivos: A Apple® implementa normas rígidas de segurança para construções de aplicativos para o iOS™. Um aplicativo por si só, tem acesso apenas a uma área restrita destinada a ele para manipulação de arquivos auxiliares. Essa área restrita é denominada *sandbox* da aplicação. O termo *sandbox* pode ser interpretado como uma caixa de areia, na qual cada aplicação poderá "brincar" somente em sua respectiva caixa de areia. Esse conceito é utilizado no iOS™ como meio de prevenir o acesso indevido de uma aplicação a outros dados do dispositivo.

  O iOS™ dispõe de um mecanismo bastante útil para o armazenamento de informações de configurações. Esse mecanismo é conhecido como arquivo com lista de propriedades. Grande parte dos aplicativos que armazenam informações preferenciais de usuários utiliza a lista de propriedades. A utilização desse mecanismo é indicada em situações que o aplicativo necessita armazenar um volume relativamente pequeno de dados.

- SQLite™: É uma biblioteca que implementa um banco de dados transacional autocontido e que não necessita de servidor para ser utilizado, pois embarca toda sua lógica em um arquivo que pode ser empacotado em sua aplicação. Para a manipulação de grandes volumes de dados nos dispositivos móveis, é possível a utilização de banco de dados e a linguagem SQL, que foi adaptada para esses dispositivos em questão. O SQLite™ é a uma versão do banco de dados SQL que geralmente é utilizado em dispositivos móveis, como *smartphones* e *tablets*. Essa versão é mais leve que o SQL e é executada de forma mais rápida e simples nesse tipo de dispositivo. Essa biblioteca é mais utilizada em aplicações que desejam independência de um servidor para armazenamento de dados, sendo desenvolvida utilizando-se da linguagem C (SQLITE, 2012).

  O SQLite™ é nativo no iOS™, não sendo necessária instalação de componentes e bibliotecas adicionais. Esse banco de dados, além da sua rapidez, consome pouca memória e é muito confiável. A manipulação direta do SQLite™ não é trivial, exigindo muita codificação.
- *Core Data*: Biblioteca disponibilizada pelo iOS™ que fornece outro mecanismo de persistência de dados. Essa biblioteca utiliza o SQLite™ como base para realizar as operações de persistência, porém torna o processo mais transparente para o desenvolvedor, o auxiliando e facilitando a implementação. As principais funcionalidades de um SGBD são fornecidas no *Core Data*. Através dessa biblioteca é possível criar tabelas de forma gráfica sem a necessidade de utilizar instruções SQL, o que torna a criação e gerenciamento do banco de dados tarefas mais intuitivas e rápidas.

2.5 ARQUITETURA MVC (*MODEL-VIEW-CONTROLLER*)

Segundo MARZULLO (2012), um estilo arquitetural determina uma abordagem para composição estrutural de um sistema. Ele define um padrão com o qual o arquiteto deve se orientar para compor seus componentes e então conectá-los. O MVC é um estilo arquitetural que é dividido em três camadas: *model*, *view* e *controller*. Essa divisão visa separar as regas de negócio da interface gráfica. Consequentemente, facilita a manutenção do código e as alterações da interface gráfica da aplicação. A Figura 5 ilustra o modelo MVC destacando a divisão das camadas e a comunicação direta entre elas.

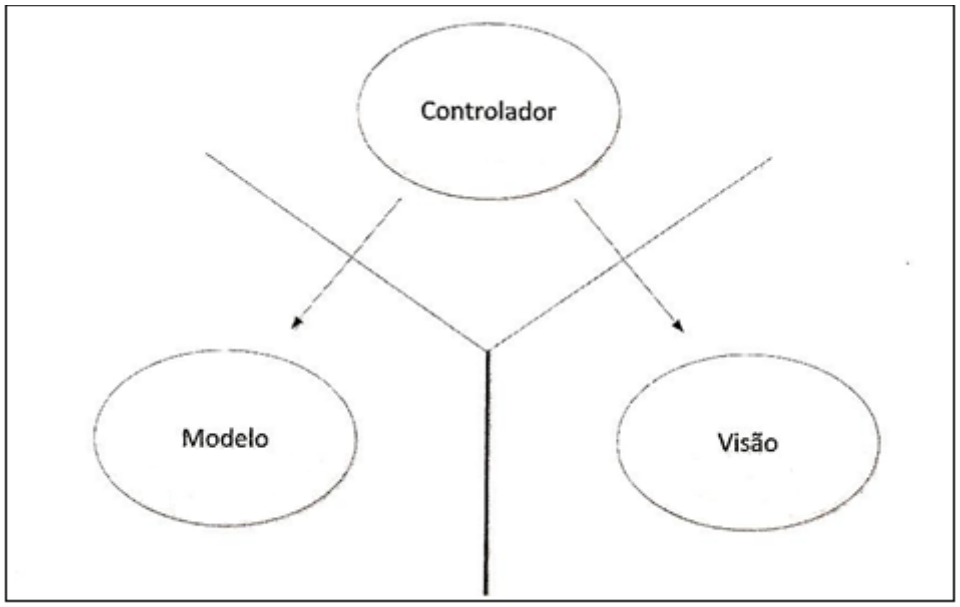

Figura 5 - Modelo MVC (MARZULLO, 2012)

- *Model*: A camada *Model* é responsável por codificar e organizar as regras de negócio da aplicação, sem se preocupar em como os dados serão apresentados para o usuário. Definições de classes, processamento de dados, persistência de dados e qualquer ação relacionada à regra de negócio são tratadas nessa camada.
- *View*: A camada *View* é responsável por apresentar informações na tela. Essas informações são extraídas da camada *Model*. Nessa camada não se deve codificar funções, processar dados nem armazenar informações. Por meio da exibição de componentes gráficos os usuários podem interagir com a aplicação.
- *Controller*: Essa camada é responsável pela comunicação entre as camadas *Model* e *View*, não devendo ser responsável pelas regras de negócio, nem por apresentar dados na tela. A camada *Controller* se comunica com as duas outras camadas, recebe requisições do usuário por meio da interface gráfica da camada *View* e envia para camada *Model* realizar algum processamento.

A biblioteca *Cocoa Touch* foi construída baseada no MVC. No iOS™ esse modelo possui algumas particularidades. Conforme ilustrado na Figura 5, o modelo MVC possui uma divisão entre as camadas, caracterizando que as camadas *Model* e *View* não devem "se conhecer", sendo a comunicação entre as duas camadas responsabilidade da camada *Controller*.

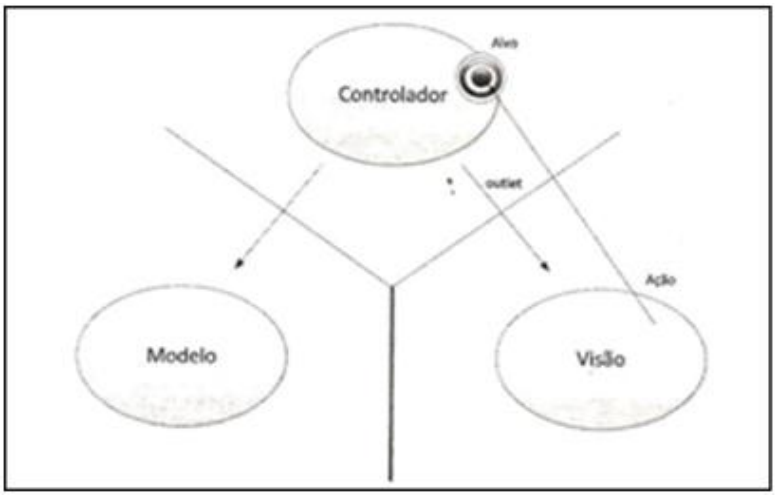

Figura 6 - Modelo MVC no iOS™ (MARZULLO, 2012)

Na Figura 6 podem-se notar novos elementos na composição do modelo, sendo eles, os *outlets* (ações) e os *targets* (alvos). Segundo MARZULLO (2012), são elementos abstratos que oferecem formas de conexão fracamente acopladas entre as camadas *model*, *view* e *controller*. Um *outlet* é um canal de comunicação que liga a camada *controller* a um componente da camada *view*, como um botão, por exemplo. Já um *target* é um ponto fixado na camada *controller* que é passado para a camada *view* para notificá-la de uma interação com o usuário.

2.6 INTERAÇÃO HUMANO-COMPUTADOR

A área de IHC se dedica a estudar os fenômenos de comunicação entre pessoas e sistemas computacionais que está na interseção das ciências da computação e informação e ciências sociais e comportamentais envolvendo todos os aspectos relacionados com a interação entre usuários e sistemas. A pesquisa em IHC tem por objetivo fornecer explicações e previsões para fenômenos de interação usuário-sistema e resultados práticos para o projeto da interação (SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 2012). Um dos principais conceitos estudados em IHC é a usabilidade, que aborda os aspectos envolvendo a comunicação entre usuário e máquina. Usabilidade pode ser entendida como uma medida na qual um produto pode ser usado por usuários específicos para alcançar objetivos específicos com eficácia, eficiência e satisfação em um contexto específico de uso (ISO 9241, 2002).

O sucesso de um software geralmente é dependente da sua usabilidade. Um software com uma interface gráfica complexa e mal projetada dificilmente irá alcançar a satisfação dos usuários. Para softwares desenvolvidos para plataforma móvel, a responsabilidade de desenvolver uma interface gráfica bem estruturada e agradável aumenta, pois na maioria das vezes, os dispositivos móveis possuem dimensões de tela consideravelmente menores, o que dificulta a interação do usuário.

Segundo NEIL (2012) as reclamações mais comuns que resultam em avaliações negativas dos usuários de aplicativos para dispositivos móveis são: travamentos, falta de recursos importantes, navegação ruim e design confuso de interface. Os dois últimos problemas podem ser evitados adotando padrões de *design*. Padrões de *design* comuns facilitam o acesso aos principais recursos da aplicação, aumentando a satisfação do usuário. Aplicativos com boa navegação são intuitivos e facilitam a realização das tarefas, desde as mais simples até as mais complexas.

3 TRABALHOS RELACIONADOS

Com o avanço tecnológico, os dispositivos móveis permitem acessar informações a qualquer hora e lugar. Devido à mobilidade e praticidade desses dispositivos, o mercado desse segmento vem crescendo substancialmente a cada ano.

TONIN (2012) realizou um trabalho destacando a evolução da computação móvel e suas tendências. Segundo a autora, a computação móvel representa um novo paradigma computacional, que ganha destaque principalmente devido à mobilidade. A computação móvel permite a comunicação sem fio, eliminando a dependência de uma conexão a uma estrutura física. O trabalho apresenta pesquisas relacionadas ao provável crescimento do segmento nos próximos anos, destacando o número de empregos gerados, investimento na área e a importância econômica da computação móvel para o desenvolvimento dos países. Além disso, outra pesquisa abordada evidencia o aumento do número de usuários e a mudança na forma de utilização desses dispositivos. Os jogos, que há dois anos eram os aplicativos mais utilizados em *smartphones,* foram ultrapassados por aplicações de sites utilizados diariamente, como redes sociais e sites de compra. O número de *smartphones* utilizando a Internet teve um crescimento de 45% desde 2010 até 2012. TONIN (2012) acredita que além do impacto causado no mercado mundial, a computação móvel tem um futuro longo e promissor. Esse paradigma permite pessoas e empresas interagirem, gerando assim novas oportunidades de negócios e sendo responsável pela geração de milhões de empregos pelo mundo, fazendo com que empresas não tenham outra escolha a não ser se adaptar a este segmento.

OLIVEIRA e COSTA (2011) desenvolveram um sistema automatizado de relacionamento com o cliente para plataforrma móvel. O sistema, batizado de Trade CRM, foi construído para o sistema operacional iOS™, com interface exclusiva para o tablet iPad®. Tiveram como principal objetivo explorar os recursos inovadores disponibilizados pelo iOS™ e escolheram o iPad® principalmente pelo grande número de usuários do dispositivo.

O sistema faz a integração com um banco de dados central quando conectado a Internet, permitindo a sincronização dos dados. Essa sincronização foi viabilizada através da construção de um *Web Service*, desenvolvido na linguagem C#. Os autores utilizaram a linguagem de programação Objective-C, banco de dados SQLite™ na aplicação móvel e SQL Server no servidor. Na aplicação móvel, é possível alterar e incluir informações, porém não é

permitida a exclusão de dados, por questão de segurança e integridade com a base central. As informações consideradas mais importantes, como saldo de estoque, histórico de compras, orçamento e lista de preços são atualizadas automaticamente sempre que o usuário acessa esses dados. Esta estratégia foi adotada pelas constantes alterações dessas informações, visando obter uma maior confiabilidade das informações. A sincronização pode ser automática ou através de uma requisição do usuário, que pode optar por sincronizar parcialmente ou totalmente as informações. Na primeira opção, apenas os dados alterados ou incluídos são sincronizados e, na segunda, a base de dados local é apagada e a base central é copiada inteiramente para o dispositivo móvel. A sincronizção total está disponível apenas para os gerentes do sistema e normalmente é utilizada na instalação do sistema em um novo dispositivo.

O sistema possibilita uma visualização no formato retrato ou paisagem de acordo com a orientação física do dispositivo. A interface tem como apoio central uma lista de opções de acesso rápido para o usuário. As opções da lista foram agrupadas por função e podem ser habilitadas ou desabilitadas conforme o perfil do usuário. Foi implementado também um sistema de busca para facilitar o acesso as informações. A Figura 7 ilustra a pesquisade clintes no sistema Trade CRM.

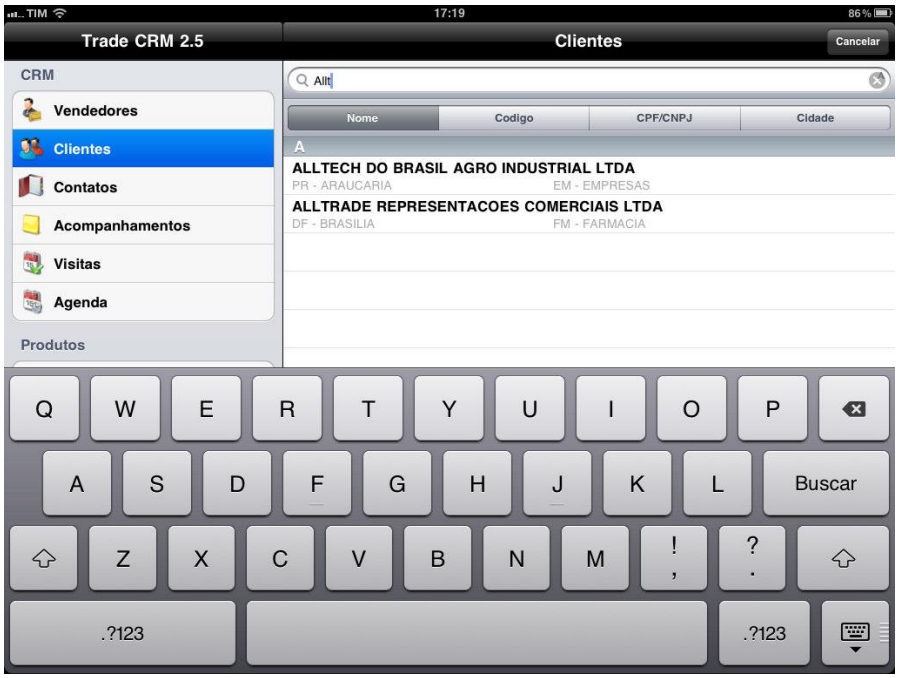

Figura 7 - Pesquisa de Clientes (OLIVEIRA e COSTA, 2011)

OLIVEIRA e COSTA (2011) fizeram uma pesquisa no mercado brasileiro, buscando casos de sucesso na implementação de sistemas de CRM que envolvesse a tecnologia móvel. Relataram o caso da empresa Cyrela Brazil Realty que adotou o sistema de CRM Dynamics da empresa Microsoft® e posteriormente revitalizou seu sistema com o uso do iPad®. Considerada uma das maiores empresas incorporadoras de imóveis do Brasil, a Cyrela Brazil Realty substituiu o software de CRM desenvolvido internamente pelo sistema da Microsoft®. Essa mudança objetivou possuir uma visão mais detalhada dos clientes atuais e potenciais, focando principalmente em suas necessidades e preferências. A empresa vê na melhor gestão dessas informações um fator que contribui para o aumento qualitativo em eficiência operacional e quantitativo para as vendas. Foram mapeadas as demandas da empresa e após customizações realizadas na ferramenta original, o sistema entrou em produção para possibilitar um melhor gerenciamento das informações. O sistema possibilitou uma maior facilidade em acompanhar o relacionamento com o cliente além de diminuir o tempo de resposta a e-mails, visto que o sistema é integrado ao sistema de mensagens eletrônicas da Microsoft®.

Após a implantação do sistema, a Cyrela começou a utilizar o iPad® para complementar a força de vendas dos imóveis. A empresa utiliza aplicativos que dispõem de informações relevantes dos clientes que incluem vídeos, plantas, localização, mapas com rede de serviços, entre outras informações. O iPad® foi colocado à disposição de corretores e clientes com o intuito de propiciar uma nova experiência de compra e venda. O dispositivo complementa os catálogos e folhetos de produtos. Porém a tendência é que gradativamente substituam os materiais impressos, tornando os meios digitais cada vez mais utilizados.

O sistema desenvolvido por OLIVEIRA e COSTA (2011) foi implantando em uma empresa que atua na distribuição, armazenamento e transporte de vacinas humanas e produtos médico-hospitalares refrigerados. Inicialmente, a empresa adquiriu cinco iPads® equipados com chips de telefonia 3G. Os dispositivos foram disponibilizados para uma equipe de vendas externas de uma nova filial. No primeiro mês da utilização do sistema foram executadas várias customizações mediante a necessidade dos vendedores. Após esse período o sistema adquiriu estabilidade e começou a operar normalmente atendendo as necessidades da empresa. A base de dados contém um grande volume de informações de clientes, visitas, contatos, históricos, dentre outras. A instalação do sistema em novo dispositivo necessita de uma sincronização total entre a base de dados local e o servidor central. Devido ao grande volume de dados utilizando a conexão 3G, o processo dessa sincronização demora

aproximadamente uma hora. Apesar do tempo de sincronização ser alto, esta não é uma questão crítica, visto que a sincronização total ocorre geralmente apenas na primeira execução do software, além disso, pode-se utilizar uma conexão com maior velocidade, como uma conexão de banda larga.

O sistema obteve boa aceitação por parte da empresa. Segundo o gerente geral, o iPad® passa uma visão positiva aos clientes, mostrando que a empresa se preocupa com a rapidez e a confiabilidade na troca de informações. A gerente de vendas, Fabíola Soldado demonstra satisfação com a utilização do novo sistema: "*Antes nós atendíamos somente os clientes do Rio Grande de Sul utilizando um netbook, o processo era extremamente lento, pois tínhamos que ligar o aparelho, esperar ele iniciar, conectar a placa de conexão, e depois acessar o sistema da empresa remotamente, o que tomava muito tempo, quase inviabilizando o trabalho. Além de um processo extremamente trabalhoso, passávamos um ar de amadorismo da empresa perante o cliente*" (OLIVEIRA e COSTA, 2011).

O tráfego de informações via Internet, foi um ponto destacado pelos autores. Muitas localidades não dispõem de um bom sinal de comunicação e uma velocidade adequada. Com isso, torna-se necessário realizar uma análise criteriosa para decidir quais informações estarão disponíveis no dispositivo móvel, visto que a cada nova informação maior será o tráfego.

4 METODOLOGIA

O projeto foi construído seguindo algumas etapas e utilizando recursos que, em conjunto, resultaram no software que será utilizado para realizar as Operações Barter da empresa ABC.

4.1 ETAPAS

As etapas do projeto foram realizadas respeitando-se uma ordem cronológica. Essa divisão de etapas sequenciais teve como finalidade evitar grandes alterações nas etapas finais que poderiam comprometer o projeto, ocasionando atraso. A Figura 8 representa o fluxograma das etapas do projeto que serão detalhadas a seguir:

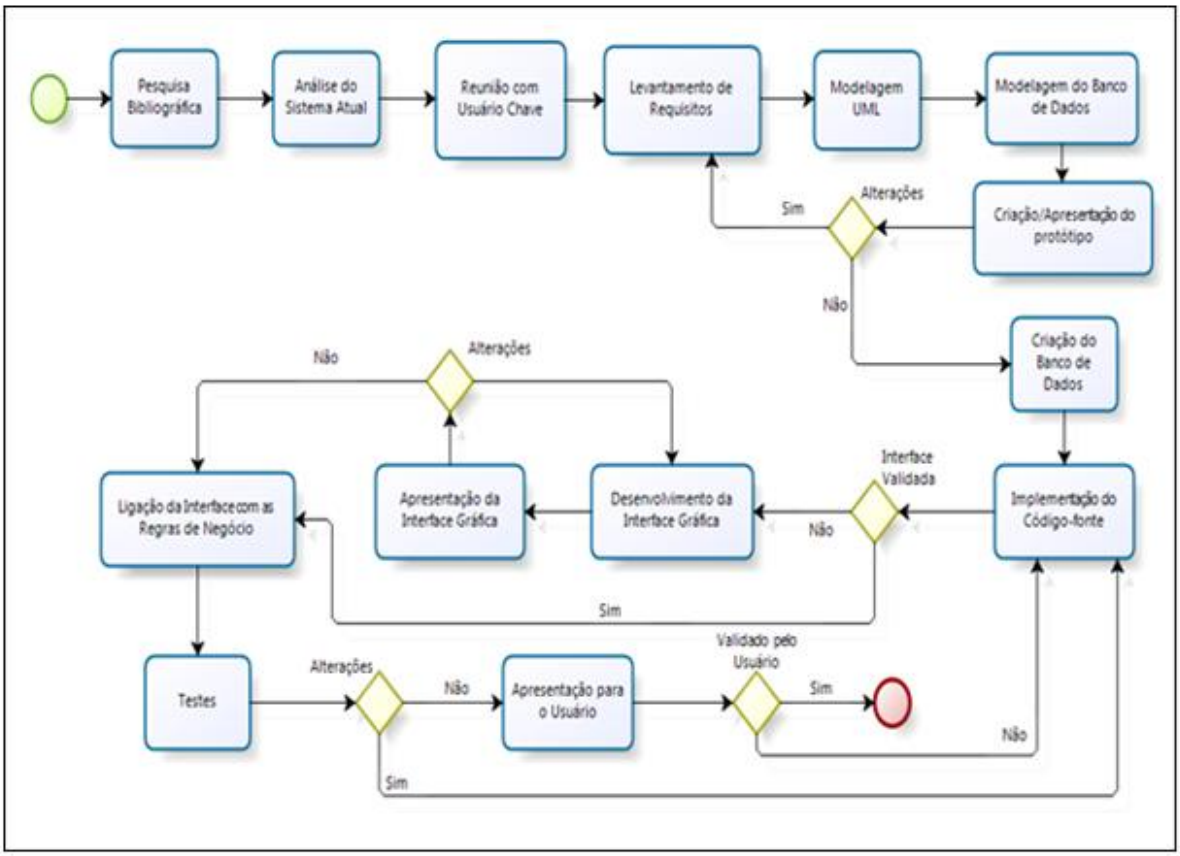

Figura 8 - Fluxograma das Etapas

- Pesquisa Bibliográfica: Para fundamentar a parte teórica do projeto foi realizada uma pesquisa bibliográfica, focada principalmente na linguagem de programação Objective-C e nas boas práticas para construção de uma interface gráfica para dispositivos móveis.
- Análise do Sistema Atual: Para compreender melhor o funcionamento do processo das Operações Barter da empresa, foi realizado um estudo de caso do sistema utilizado atualmente na empresa.

  Reunião com Usuário-Chave: Os itens identificados na etapa anterior foram apresentados para o usuário-chave (coordenador de operações comerciais da empresa), que definiu a viabilidade e a necessidade das melhorias sugeridas.
- Levantamento de Requisitos: Foi realizado um levantamento de requisitos em conjunto com o usuário-chave. Nessa etapa foram definidas as funcionalidades do novo software e as regras de negócio utilizadas.
- Modelagem UML (*Unified Modeling Language*): Após identificar os requisitos do sistema, foi utilizada a linguagem de modelagem UML para modelar o Diagrama de Caso de Uso e o Diagrama de Classes.
- Modelagem do Banco de Dados: A modelagem do banco de dados do sistema foi construída com base no Diagrama de Classes construído na etapa anterior. Nessa etapa foram definidas tabelas, campos, relacionamentos entre as tabelas, além das chaves primárias e estrangeiras.
- Apresentação do Protótipo: Após a modelagem dos diagramas e do banco de dados, foi desenvolvido e apresentado um protótipo para o usuário chave com a finalidade de identificar possíveis modificações.
- Criação do Banco de Dados: Seguindo os conceitos da Arquitetura MVC, o desenvolvimento do sistema foi realizado separando as regras de negócios da interface gráfica. Nessa etapa, o Banco de Dados do sistema foi criado utilizando os recursos da biblioteca SQLite™. O banco foi construído com base na modelagem feita anteriormente.
- Implementação do Código Fonte: Nessa etapa foram codificadas as classes do sistema com base no Diagrama de Classes e as regras de negócio com base no Diagrama de Casos de Uso e nos requisitos levantados nas etapas anteriores.
- Desenvolvimento da Interface Gráfica: Nessa etapa foi construída a interface gráfica do software. O projeto da interface foi realizado seguindo padrões de *design* para

dispositivos móveis e conceitos de IHC, focando principalmente na usabilidade. Com isso foi possível criar uma interface simples e intuitiva.

- Apresentação da Interface Gráfica: A interface gráfica foi apresentada e aprovada pelo usuário.
- Ligação da Interface com as Regras de Negócio: Com a Interface Gráfica validada pelo usuário, os componentes gráficos foram conectados com o código fonte.
- Testes: Foram realizados testes para avaliar o desempenho, facilidade de uso e integridade dos dados. Os teste foram realizados no iOS Simulator e no iPad®.
- Apresentação para o Usuário: Após a conclusão dos testes, o software foi apresentado para o usuário-chave sendo executado no dispositivo iPad®.

4.2 RECURSOS

Para o desenvolvimento do sistema foi utilizado um MacBook® com o sistema operacional Mac OS®, no qual foi utilizada a IDE Xcode.

Para construção dos diagramas UML e modelagem do banco de dados foram utilizadas as ferramentas gratuitas StarUML e DBDesigner. Essas ferramentas foram utilizadas em um notebook Dell™ com o sistema operacional Windows 7®.

Para a realização de testes, foi utilizado um iPad® modelo 3ª geração com o sistema operacional iOS™ 6.1, além do iOS Simulator. Para visualizar os registros gravados no banco de dados foi utilizado o software *SQLite Data Base Browser*, disponível para o Mac OS®.

5 DESENVOLVIMENTO

Neste capítulo serão abordadas as etapas de todo o desenvolvimento do projeto.

5.1 ANÁLISE DO SISTEMA ATUAL

Foi realizada uma análise do sistema Barter utilizado na empresa atualmente. A análise foi feita, a priori, no software em funcionamento, na qual foram realizadas as operações rotineiras dos usuários do sistema. Posteriormente foi analisado o código-fonte e o banco de dados do sistema, além de aspectos referentes à conexão com a Internet e sincronia de dados.

No sistema em funcionamento não foram identificadas falhas, porém a interface gráfica não ofereceu uma experiência agradável. Alguns componentes gráficos estão mal localizados e em alguns momentos a tela fica com excesso de informações, prejudicando o entendimento do usuário e dificultado as operações. As mensagens de avisos e erros funcionaram corretamente e foram suficientes para o entendimento das interações com o sistema.

O código-fonte está bem comentado e estruturado, o que facilita o entendimento das classes, e regras de negócio. A estrutura do banco não foi construída da maneira correta, pois as relações entre as tabelas não foram criadas utilizando boas práticas, podendo, em alguns momentos ocasionar em inconsistências de informações.

Essa análise foi produtiva, pois antes da reunião com o usuário-chave para o levantamento de requisitos foi essencial ter uma ideia do processo e da maneira com que os usuários trabalham, além de ter em mente a estrutura de tabelas e classes.

5.2 LEVANTAMENTO DE REQUISITOS

Foi realizada uma reunião com o usuário-chave, na qual foram abordadas as necessidades do sistema, ideias para o novo sistema, aspectos importantes do sistema atual,

entre outros assuntos relevantes ao projeto. A partir dessa reunião foram especificados os requisitos funcionais e não funcionais detalhados a seguir.

5.2.1 REQUISITOS FUNCIONAIS

Os requisitos funcionais do sistema definem as funcionalidades que um sistema deve dispor. O sistema para Operações Barter deve conter:

- RF001: O sistema deverá possibilitar ao usuário a consulta de clientes.
- RF002: O sistema deverá possibilitar ao usuário a consulta de produtos.
- RF003: O sistema deverá possibilitar ao usuário a consulta de cotações.
- RF004: O sistema deverá possibilitar ao usuário a consulta de simulações.
- RF005: O sistema deverá possibilitar ao usuário a inserção de simulações.
- RF006: O sistema deverá possibilitar ao usuário a exclusão de simulações.

5.2.2 REQUISITOS NÃO FUNCIONAIS

Os requisitos não funcionais do sistema definem as tecnologias utilizadas e características do sistema. O sistema para Operações Barter deve conter:

- RNF02: O sistema deverá ser suportado pelo sistema operacional iOS™ 6.1 ou superior.
- RNF03: O sistema deverá ser implementado na linguagem Objective-C.
- RNF04: O sistema deverá ser construído, optando quando possível, por componentes gráficos nativos do sistema operacional iOS™ e já utilizados em aplicações nativas.
- RNF05: O sistema deverá oferecer componentes gráficos intuitivos para facilitar a interação com o usuário.
- RNF06: O sistema deverá suportar somente a orientação paisagem.
- RNF07: O sistema deverá oferecer informações claras e objetivas.
- RNF08: O sistema deverá realizar as operações solicitadas pelo usuário de maneira correta.
- RNF09: O sistema deverá estar disponível em qualquer ambiente, mesmo quando não houver conexão com a Internet.

- RNF10: O sistema deverá ser construído de forma a se tornar fácil a localização de falhas.
- RNF11: O sistema deverá estar preparado para futuras customizações nas regras de negócios.
- RNF12: O sistema deverá estar preparado para futuras customizações na interface gráfica.

5.3 MODELAGEM UML

Com base na análise do sistema atual, as informações coletadas na reunião e os requisitos funcionais, foram construídos os diagramas de Caso de Uso e de Classes utilizando o software o StarUML™.

5.3.1 DIAGRAMA DE CASO DE USO

O sistema que será utilizado em um dispositivo móvel é parte de um projeto maior que será responsável por gerir as Operações Barter na empresa. O escopo do projeto engloba somente a parte cliente (estrutura cliente/servidor). Dessa maneira, muitas informações estarão disponíveis apenas para consulta no sistema móvel, não sendo possível o usuário alterar, incluir ou excluir informações, sendo essas operações, de responsabilidade do servidor. Futuramente os dados serão lidos pelo sistema móvel através de um *Web Service*, fazendo a leitura de arquivos no formato XML (*eXtensible Markup Language*). O Diagrama de Caso de Uso foi modelado conforme Figura 9.

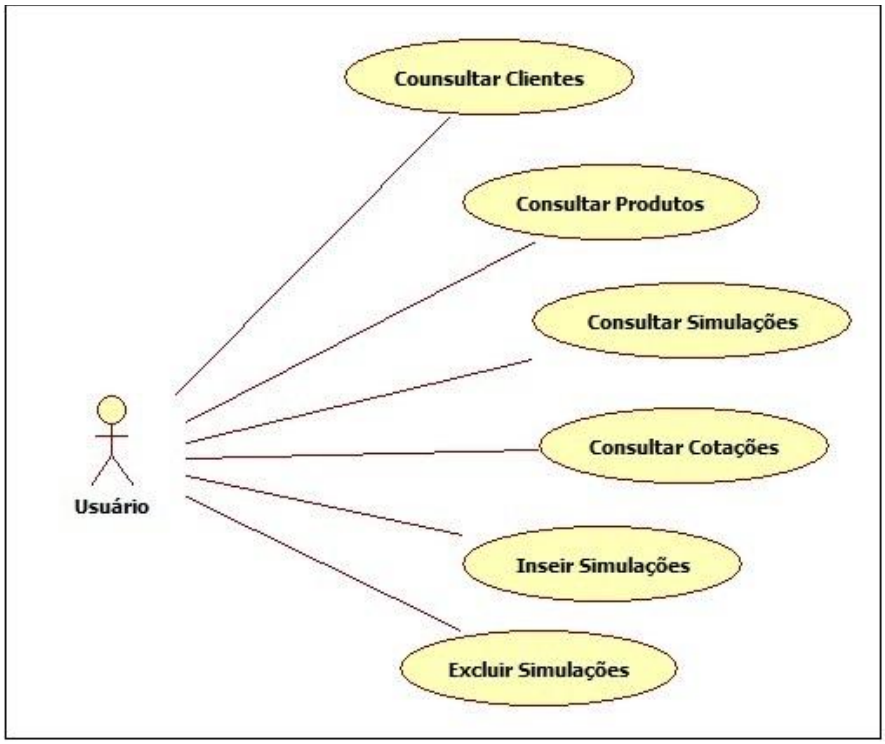

Figura 9 - Diagrama de Caso de Uso

5.3.2 DIAGRAMA DE CLASSES

O Diagrama de Classes foi construído baseado no sistema atual utilizado na empresa, visto que as classes estão bem estruturadas e as regras de negócios não sofrerão alterações significativas para o novo sistema. O modelo foi construído conforme ilustrado na Figura 10.

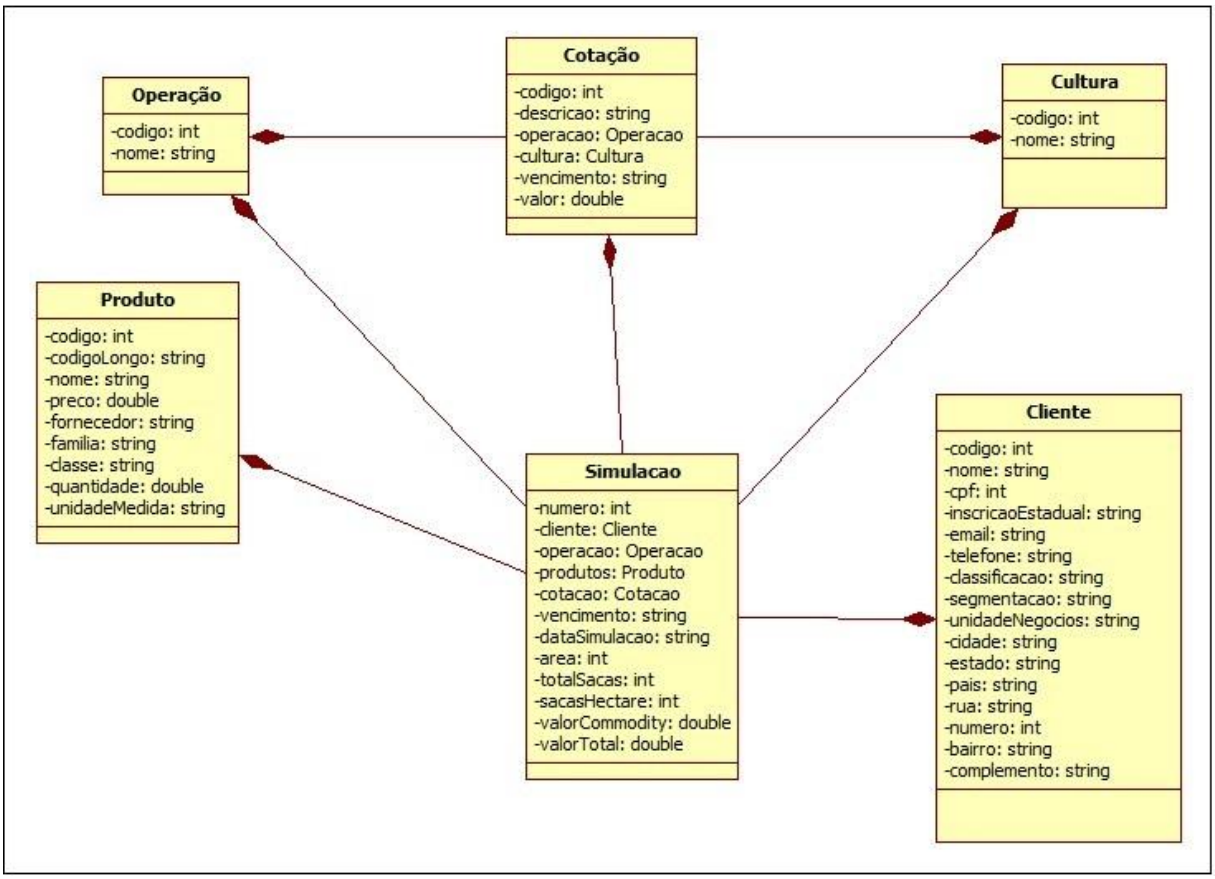

Figura 10 - Diagrama de Classes

5.4 MODELAGEM DO BANCO DE DADOS

O banco de dados é uma das partes mais importantes para a construção do software. A má estruturação do banco de dados pode acarretar muitos problemas no futuro e até mesmo resultar em um fracasso de todo o projeto.

Para minimizar os possíveis problemas futuros foi criado o modelo de entidade-relacionamento com base na estrutura do atual, fazendo melhorias significativas principalmente nos relacionamentos entre as tabelas. Também foi utilizado como base o Diagrama de Classes apresentado na seção 5.3.2.

Para a construção do diagrama foi utilizado o software *DBDesigner* na versão 4.0. O diagrama de entidade-relacionamento é ilustrado na Figura 11.

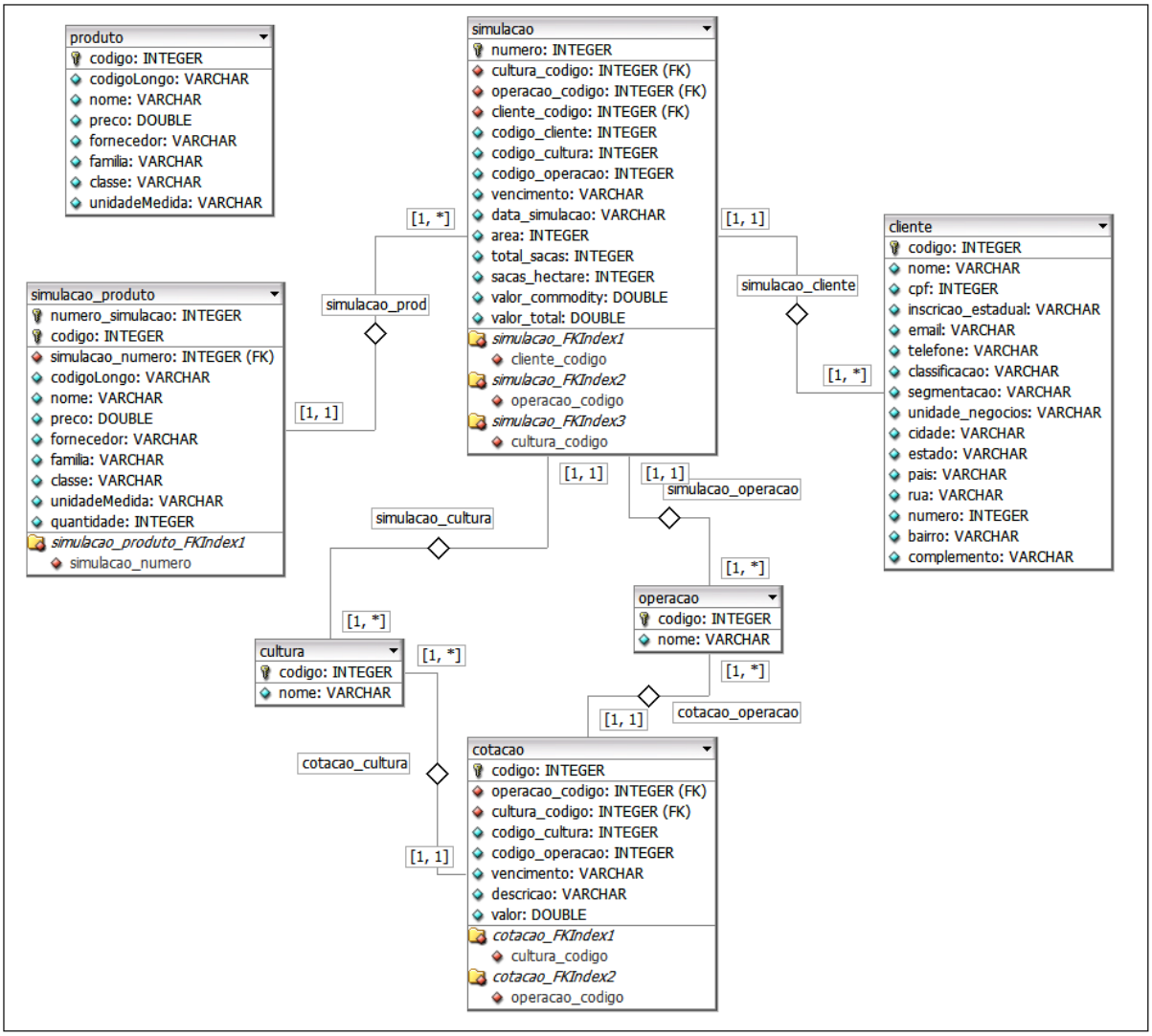

Figura 11 - Diagrama Entidade-Relacionamento

5.5 CONSTRUÇÃO DO BANCO DE DADOS

Após alguns testes práticos utilizando o SQLite™ e o *Core Data* foi feita a escolha pela primeira opção. Apesar do *framework Core Data* proporcionar um desenvolvimento mais rápido, pois possibilita a construção do banco de dados de maneira gráfica, o SQLite™ propicia um controle maior das operações, tornando mais fácil a manutenção do código-fonte.

Dessa maneira, o banco de dados foi construído seguindo o modelo de entidade-relacionamento apresentado na seção 5.4. Todas as tabelas e operações do banco de dados foram construídas via código-fonte, não utilizando nenhum editor gráfico. Para visualizar os

dados e realizar os testes no banco de dados foi utilizado o software *SQLite Data Base Browser*, disponível para a plataforma Mac OS®.

O trecho de código, ilustrado na Figura 12, representa um método da classe *CotacaoDB*, que é responsável por retornar um vetor de objetos da classe *Cotacao* cadastrado no banco de dados.

```
- (NSMutableArray *) getCotacao{
    NSMutableArray *cotacoes = [[NSMutableArray alloc] init];

    NSString *query = @"select _id, codigoCultura,codigoOperacao,
        descricao,vencimento,valor from cotacao order by codigoOperacao";

    sqlite3_stmt *stmt;

    int resultado = sqlite3_prepare_v2( bancoDeDados, [query UTF8String],-1, &stmt, nil);
    if (resultado == SQLITE_OK) {

        while (sqlite3_step(stmt) == SQLITE_ROW) {
            Cotacao *cotacao = [Cotacao alloc];

            cotacao._id = sqlite3_column_int(stmt, 0);

            cotacao.Cultura  = [GetData getCultura  :sqlite3_column_int(stmt, 1)];
            cotacao.operacao = [GetData getOperacao :sqlite3_column_int(stmt, 2)];

            char *s = (char *)sqlite3_column_text(stmt, 3);
            cotacao.descricao = s != nil ? [NSString stringWithUTF8String:s] : nil;


            s   = (char *)sqlite3_column_text(stmt, 4);
            cotacao.vencimento = s != nil ? [NSString stringWithUTF8String:s] : nil;

            s   = (char *)sqlite3_column_text(stmt, 5);
            cotacao.valor = s != nil ? [[NSString stringWithUTF8String:s] doubleValue] :
                0;

            [cotacoes addObject:cotacao];
        }
        sqlite3_finalize(stmt);
    }
    return cotacoes;
}
```

Figura 12 - Trecho de Código da Classe CotacaoDB

A variável *query*, do tipo texto, recebe o comando *SQL* que irá retornar todos os registros contidos na tabela *cotacao*. Esse comando é executado e se não retornar erros, é iniciado um *loop* que é finalizado somente quando todos os registros da tabela forem lidos. A cada registro lido, um objeto do tipo *Cotacao* é instanciado e suas variáveis recebem os respectivos valores gravados no banco de dados. Ao fim da leitura de cada registro, o objeto do tipo Cotacao é adicionado a um vetor. Ao final da leitura de todos os registros da tabela, o método retorna o vetor contendo todos os objetos do tipo Cotacao.

5.6 IMPLEMENTAÇÃO

Primeiramente foram criadas as classes de acordo com os Diagramas de Classes e Entidade Relacionamento.

Para o desenvolvimento da interface gráfica, foi realizado um *Benchmarking* de aplicações desenvolvidas para iPad® com algumas características similares a aplicação proposta neste projeto. Através da análise dessas aplicações foram observadas várias características visuais positivas e negativas. Tratando-se do desenvolvimento para plataforma móvel, a usabilidade torna-se um fator ainda mais importante para o sucesso do software. Nas avaliações das aplicações analisadas, grande parte das críticas e elogios feitos pelos usuários remete a interface gráfica e facilidade de uso. Com base nessas análises e dos requisitos não funcionais, a interface gráfica foi projetada para facilitar ao máximo a utilização do sistema pelo usuário. Para tornar o ambiente familiar ao usuário foram utilizados componentes gráficos nativos do sistema, reduzindo assim o período de adaptação do usuário com o novo sistema.

Durante todo o desenvolvimento do projeto foi utilizado o modelo arquitetural MVC, dividindo o software em três camadas. Essa divisão visou separar as regas de negócio da interface gráfica. Desta forma, a manutenção do código se torna uma tarefa mais fácil, além de possibilitar uma rápida customização da interface gráfica sem que haja alterações nas demais camadas.

Após implementar uma funcionalidade do sistema, a mesma era apresentada ao usuário-chave que era responsável por avaliar a interface gráfica e as regras de negócios. Quando este sugeria uma melhoria, inicialmente era analisada a viabilidade e, quando possível, a solicitação era atendida.

5.7 TESTES

Os testes do sistema ocorreram durante toda a fase de desenvolvimento da aplicação, porém foram intensificados após todas as funcionalidades estarem concluídas. Inicialmente foi realizado um teste de facilidade de uso com os futuros usuários do sistema.

Obtiveram-se resultados positivos com os testes, visto que os usuários não encontraram dificuldades para realizar as operações e classificaram o software como fácil de usar.

A incidência de erros é mínima, pois todos os campos preenchidos pelo usuário são selecionados dos respectivos cadastros (com exceção da *Área* e *Quantidade do Produto*). Com isso, o usuário não necessita digitar para preencher as informações, o que impede de digitar algum dado inconsistente. Para os campos de *Área* e *Quantidade do Produto*, somente são aceitos dígitos numéricos. Se o usuário acidentalmente digitar algum caractere alfanumérico ou especial, essa entrada é simplesmente ignorada pelo sistema. Além disso, o botão de confirmação somente é ativado quando o usuário preencher um número maior que zero. Durante os testes foram identificados pequenos problemas que foram sanados sem muitos ajustes.

5.8 APRESENTAÇÃO DO SOFTWARE

A seguir serão apresentadas algumas figuras ilustrando o programa em funcionamento no iPad® modelo 3ª geração.

A Figura 13 ilustra a tela inicial da aplicação, exibindo os ícones que permitem ao usuário acessar os recursos do sistema. O ícone *Simulações* direciona o usuário para uma tela na qual o usuário poderá consultar, remover e adicionar simulações. Os ícones *Clientes, Produtos, Cotações* direcionam o usuário para as telas de consulta de suas respectivas descrições. O ícone *Sobre* direciona o usuário para uma tela em que o mesmo pode visualizar informações sobre o software, além de poder entrar em contato com o desenvolvedor do sistema.

Foram utilizados ícones com um tamanho grande para facilitar a interação do usuário com as funcionalidades do sistema e um fundo de tela ligado ao setor de agronegócios.

Figura 13 - Tela Inicial

Após selecionar o ícone *Clientes* é exibida a tela para consultar clientes, a qual é ilustrada na Figura 14. Para construção dessa tela foi utilizada uma estrutura gráfica denominada de *splitView*. Nessa estrutura, a tela é dividida em duas visões distintas e independentes. A visão do lado esquerdo da tela é denominada *MasterView*, enquanto a visão do lado direito é denominada *DetailView*. Esse tipo de estrutura é comumente utilizada em vários aplicativos, inclusive nas aplicações nativas do iPad®, como a aplicação de configurações, por exemplo.

Esse tipo de estrutura gráfica aproveita o espaço da tela do dispositivo, na qual do lado esquerdo geralmente é utilizada uma tabela em forma de lista, e do lado direito o detalhamento dos dados.

Foi utilizado também um componente para pesquisar algum cliente pelo nome para facilitar a busca do usuário. As telas utilizadas para consultar produtos e cotações são similares a esta, utilizando a mesma estrutura e componentes.

Figura 14 - Consulta de Clientes

A principal funcionalidade do sistema é o gerenciamento de simulações. A Figura 15 ilustra a tela inicial de simulações. Nessa tela é possível excluir, visualizar e adicionar uma nova simulação, sendo que para a implementação desta, também foi utilizada a arquitetura gráfica *splitView.*

.

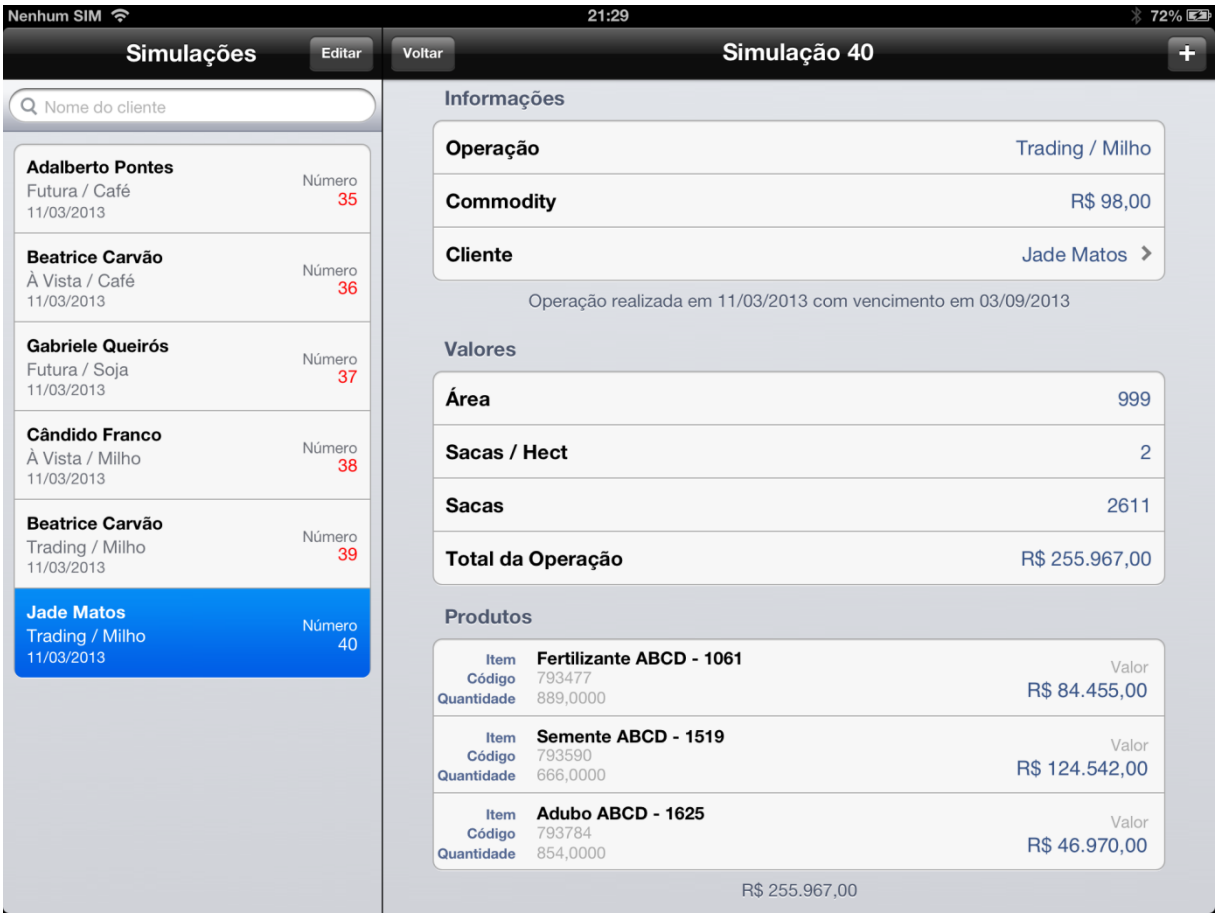

Figura 15 - Tela de Simulações

Ao selecionar uma simulação são exibidas informações da mesma do lado direito da tela. As informações apresentadas foram divididas em três seções. As duas primeiras seções são fixas, nas quais são exibidas as informações gerais e valores da simulação. Na ultima seção é exibida a listagem de produtos.

Um aspecto importante em aplicações desenvolvidas para dispositivos móveis é a possibilidade de o usuário retornar ao passo imediatamente anterior. Para suprir essa necessidade foi utilizado um botão no canto superior esquerdo com a opção de voltar. Esse tipo de abordagem é muito comum em aplicações desenvolvidas para a plataforma iOS™, pois é comum usuários se confundirem com as telas e então surgir a necessidade de voltar a tela imediatamente anterior. Foi utilizado um componente denominado *navigationController* para realizar essa transição de telas. Esse componente implementa uma estrutura de dados do tipo pilha. A cada ação do usuário de avançar ou abrir uma nova tela, a mesma é empilhada na memória, e em contrapartida, quando o usuário opta por voltar, a tela do topo é desempilhada

e descartada da memória. Utilizando essa abordagem diminui-se o risco do usuário se perder durante a execução de uma tarefa

A Figura 16 apresenta a tela na qual o usuário pode adicionar uma nova simulação no sistema. Todos os campos devem ser preenchidos pelo usuário. Caso o usuário não preencha algum campo, este é notificado através uma mensagem na tela e impede-se o avanço do processo.

Figura 16 - Nova Simulação – Informar Dados

Após o usuário preencher as informações e selecionar o botão próximo é carregada a tela ilustrada na Figura 17, na qual o usuário selecionará os produtos e visualizará algumas informações da simulação. O primeiro campo visualizado pelo usuário é a *Área* que foi informada na tela anterior. O campo *Sacas* é calculado através da divisão entre o valor total da operação (soma de todos os produtos) pelo valor da *commodity*. A divisão do número de sacas pela área de plantio é informada no campo *Sacas/hect,* que representa uma média de quantas sacas em cada hectare plantado o produtor deverá repassar a empresa.

Após selecionar os produtos desejados, o usuário poderá selecionar o botão *Concluir*, sendo notificado que a simulação foi gravada com sucesso. Após esse procedimento, as duas telas empilhadas na memória anteriormente são descartadas e a simulação já estará disponível para consulta na tela de simulação, ilustrada na Figura 15.

Figura 17 - Nova Simulação - Selecionar Produtos

6 RESULTADOS OBTIDOS

O software produzido foi aceito de maneira satisfatória pela empresa. Devido à facilidade de uso da aplicação e a utilização de um dispositivo moderno e portátil, a empresa tem o interesse de continuar investindo em soluções para tecnologias móveis, fornecendo equipamentos e investindo em cursos na área de desenvolvimento.

Nos testes, as simulações foram realizadas de maneira rápida pelos usuários, que não encontraram dificuldades em realizar as operações rotineiras no sistema. Os usuários também consideram o sistema fácil de usar e tiveram poucas dúvidas em como realizar as tarefas. O tempo de resposta também foi satisfatório, na qual todas as requisições feitas pelos usuários foram atendias em menos de dois segundos.

7 CONCLUSÃO

Com o desenvolvimento finalizado do software Barter para a plataforma iOS™, o objetivo do projeto foi concluído. Com o sistema em funcionamento é possível realizar as operações Barter da empresa de uma maneira mais simples e ágil. A utilização do iPad® irá proporcionar também a mobilidade almejada.

O sistema não será utilizado de imediato pela empresa, pois será necessária a construção da parte servidor do projeto. Porém com o banco de dados estruturado, o software está preparado para se integrar aos sistemas de ERP e BI da empresa.

A interface gráfica do software superou as expectativas dos usuários. Mesmo alguns usuários que não estavam familiarizados com o dispositivo iPad® e até mesmo com a tecnologia *touchscreen* não tiveram dificuldades em utilizar o software.

O projeto como um todo foi satisfatório pois atendeu aos requisitos, resultando em um software consistente, com uma interface gráfica agradável e fácil de utilizar.

Com os resultados obtidos com o projeto, o mesmo contribui para que as empresas possam investir mais em soluções para dispositivos móveis. Esse segmento é promissor e a tendência é de um alto crescimento a cada ano. As empresas que aderirem a soluções para esse segmento podem conseguir uma vantagem competitiva e agregar valor aos seus produtos e serviços.

8 TRABALHOS FUTUROS

Com a conclusão do software que será utilizado nos dispositivos móveis, tem-se a necessidade do desenvolvimento da parte servidor do projeto.

A aplicação servidor será desenvolvida utilizando o ERP já utilizado na empresa. Será criado um módulo para as Operações Barter que será responsável por gerenciar todas as negociações Barter da empresa. Será implementado um *Web Service* no próprio ERP, que será responsável por disponibilizar arquivos XML com os dados que serão lidos pelo dispositivo móvel.

Após a conclusão do módulo Barter no ERP, será criado um mapeamento das tabelas no sistema de BI. Após o mapeamento será possível criar relatórios gerenciais que ajudarão a alta gerencia nas tomadas de decisão.

Com a satisfação da empresa em relação ao sistema móvel, existe a possibilidade de novos projetos no segmento de dispositivos móveis para atender outras demandas da empresa, como a área de vendas e CRM.

9 BIBLIOGRAFIA

Apple. (2012a). *About the iOS Technologies*. Acesso em 02 de Maio de 2012, disponível em Apple: http://developer.apple.com/library/ios/#documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html

Apple. (2012b). *Apple - Legal - NeXT Tradmark List*. Acesso em 23 de Setembro de 2012, disponível em Apple: http://www.apple.com/legal/trademark/nexttmlist.html

Apple. (2012c). *Apple - Legal - Trademark List*. Acesso em 23 de Setembro de 2012, disponível em Apple: http://www.apple.com/legal/trademark/appletmlist.html

Apple. (2012d). *Apple Store*. Acesso em 30 de Agosto de 2012, disponível em Apple: http://store.apple.com/us

Apple. (2012e). *Especificações Técnicas do Novo iPad*. Acesso em 15 de Outubro de 2012, disponível em Apple: http://www.apple.com/br/ipad/specs/

Apple. (2012f). *iOs Technology Overiew: iOs Developer Tools*. Acesso em 18 de Outubro de 2012, disponível em Apple: http://developer.apple.com/library/IOs/#documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/iPhoneOSDeveloperTools/iPhoneOSDeveloperTools.html#//apple_ref/doc/uid/TP40007898-CH7-SW1

Apple. (2012g). *The Objective-C Programming Language*. Acesso em 12 de Setembro de 2012, disponível em Apple: http://developer.apple.com/library/mac/#documentation/Cocoa/Conceptual/ObjectiveC/Introduction/introObjectiveC.html#//apple_ref/doc/uid/TP30001163-CH1-SW2

Exame. (2012). *Android e iOS dominam mercado global de smartphones*. Acesso em 09 de Agosto de 2012, disponível em Exame: http://exame.abril.com.br/tecnologia/noticias/android-e-ios-dominam-mercado-global-de-smartphones

Governo do Estado de Minas Gerais. (2012). *Agronegócio Vigoroso e Promissor*. Acesso em 18 de Junho de 2012, disponível em Governo de Minas Gerais: http://www.mg.gov.br/governomg/portal/m/governomg/invista-em-minas/invista-em-minas/11987-agronegocio/11972/5042

IDC Brasil. (2012). *Pesquisa da IDC revela que foram vendidos mais de 370 mil tablets no primeiro trimestre de 2012*. Acesso em 27 de Junho de 2012, disponível em IDC Brasil: http://www.idcbrasil.com.br/releases/news.aspx?id=739

ISO 9241. (08 de 2002). *Requisitos Ergonômicos para Trabalho de Escritórios com Computadores.* Acesso em 08 de 11 de 2012, disponível em http://www.inf.ufsc.br/~cybis/pg2003/iso9241-11F2.pdf

Kochan, S. G. (2012). *Programming in Objective-C, Fourth Edition* (4 ed.). (M. Thurston, Ed.) Indianapolis, IN, United States of America: Addison-Wesley Developer's Library.

Lee, W.-M. (2011). *Introdução ao Desenvolvimento de Aplicativos para o Android.* (P. A. Marques, Ed.) Rio de Janeiro, RJ, Brasil: Editora Moderna Ltda.

Marzullo, F. (2012). *iPhone na Prática : aprenda passo a passo a desenvolver soluções para iOS.* (R. Prates, Ed.) São Paulo, SP, Brasil: Novatec Editora.

Milani, A. (2012). *Programando para iPhone e iPad: Aprenda a construir aplicativos para o iOS.* (R. Prates, Ed.) São Paulo, SP, Brasil: Novatec Editora.

Neil, T. (2012). *Padrões de Design para Aplicativos Móveis.* (R. Prates, Ed.) São Paulo, SP, Brasil: Novatec Editora.

Oliveira, L. d., & Costa, E. (11 de 2011). *Sistema de Gestão de Relacionamento com Clientes na Era da Mobilidade com Uso do iPad para Empresas de Comércio em Geral*. Acesso em 01 de 11 de 2012, disponível em http://www.ulbra.inf.br/joomla/images/documentos/TCCs/2011_01/TCCI_CC_LucianoSantosOliveira.pdf

Pontual, J. (2012). *Microsoft lança tablet Surface para competir com iPad da Apple*. Acesso em 21 de Agosto de 2012, disponível em Globo: http://g1.globo.com/jornal-da-globo/noticia/2012/06/microsoft-lanca-tablet-surface-para-competir-com-ipad-da-apple.html

Sociedade Brasileira de Computação. (2012). *Interação Humano Computador*. Acesso em 19 de Outubro de 2012, disponível em Sociedade Brasileira de Computação: http://www.sbc.org.br/index.php?option=com_content&view=category&layout=blog&id=45&Itemid=66

SQLite. (2012). *SQLite About*. Acesso em 10 de Agosto de 29, disponível em SQLite: http://www.SQLite.org/about.html

Taguchi, V. (Outubro de 2010). Troca Eficiênte no Campo. Acesso em 11 de 12 de 2012, disponível em Dinheiro Rural: http://revistadinheirorural.terra.com.br/secao/agrofinancas/troca-eficiente-no-campo

Tonin, G. S. (2012). *Tendências em Computação Móvel*. Acesso em 09 de 12 de 2012, disponível em http://grenoble.ime.usp.br/~gold/cursos/2012/movel/Tendencias.pdf