CARLOS ALBERTO MOTA CANDREVA

USABILIDADE DA INTERFACE METRO DO WINDOWS 8

ASSIS – SP

2013

CARLOS ALBERTO MOTA CANDREVA

USABILIDADE DA INTERFACE METRO DO WINDOWS 8

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Ciência da Computação do Instituto Municipal do Ensino Superior de Assis – IMESA e Fundação Educacional do Município de Assis – FEMA, como requisito para a obtenção do Certificado de Conclusão.

Orientador: Luiz Ricardo Begosso

Área de Concentração: Usabilidade, Programação para Windows 8

ASSIS – SP

2013

DEDICATÓRIA

Dedico este trabalho aos meus pais Carlos Alberto Candreva e Maisa Mota Candreva, por terem criado a pessoa que sou e me ajudado durante todo o curso. À minha namorada Thais

Daniele Schiavão e Souza, por toda atenção, ajuda e compreensão.

"*As pessoas que te dizem para não dar chances, todas elas perderam o sentido da vida. Você só vive uma vez, então aproveite a chance, não termine como outros dançando a mesma música.*" (Motorbreath, Metallica).

AGRADECIMENTOS

Agradeço a Deus e à minha família, por sempre terem me proporcionado tudo o que precisei para chegar até aqui.

Ao meu orientador Luiz Ricardo Begosso, e aos demais professores, por todo ensinamento transmitido ao longo da minha formação.

A todos os meus amigos e a todos que, direta ou indiretamente, contribuíram para a realização deste trabalho.

RESUMO

O trabalho apresenta conceitos da nova interface gráfica apresentada pela Microsoft, a interface Metro. Analisando a nova interface utilizando os conceitos de usabilidade de software para avaliar se a interface está adequada segundo os padrões estipulados de usabilidade verificando se ela é intuitiva e fácil de ser utilizada pelos usuários. Aplicar o conceito da usabilidade utilizando os padrões propostos da interface Metro em um aplicativo desenvolvido para Windows 8. O desenvolvimento do aplicativo deve envolve padrões de interface estabelecidos pela Microsoft, utilizando conceitos de XAML para programação da interface, C# e *WebServices* em PHP. Os *WebServices* são responsáveis por trazer as informações ao aplicativo, que são tratadas pelo C# e exibidas no aplicativo para o usuário. A usabilidade em um software ou sistema operacional é o que determina sua aceitação no mercado, pois quanto mais usuário falando bem de sua experiência de uso com o software, mais usuários estarão procurando este mesmo software, podendo assim ter uma grande adesão de mercado. Um software que tende a ser dificultoso ao usuário consequentemente não terá boas indicações, pode dar uma diferença na hora da escolha no mercado.

Palavras-chave: Usabilidade; Windows 8; WebServices, C#.

ABSTRACT

The paper presents the concepts of the new graphical interface presented by Microsoft, the Metro interface. Analyzing the new interface using the concepts of software usability to assess whether the interface is adequate according to the stipulated standards of usability checking it is intuitive and easy to use by users. Applying the concept of usability using the standards proposed Metro interface in an application developed for Windows 8. The development involves the application must interface standards set by Microsoft, using concepts of XAML for interface programming, C # and Web Services in PHP. The Web Services are responsible for bringing the information to the application which are handled by C# and displayed, in the application to the user. The usability in a software or operating system is what determines its market acceptance, as well as more user speaking of his experience with the use of software more users will be looking for the same software, so you can have a large membership market. A software that tends to be bad to the user therefore will not have good indications can make a difference when choosing the market.

Keywords: Usability, Windows 8, WebServices, C #.

LISTA DE ILUSTRAÇÕES

LISTA DE CÓDIGOS

LISTA DE ABREVIATURAS E SIGLAS

DRM: *Digital Rights Management", em português "Gestão de Direitos Digitais"*

CLR: *Common Language Runtime*

JIT: *Just In Time Compiler*

XAML: *eXtensible Application Markup Language*

JSON: *JavaScript Object Notation*

Sumário

1 INTRODUÇÃO

Com a chegada do Windows 8 e a inovação trazida com a nova interface a Microsoft quis proporcionar uma integração nas interfaces de tablets, notebooks, desktops e smartphones, surgiram certos questionamentos básicos: Até que ponto esta nova interface é realmente usual e de fácil acesso para todos os públicos? Mesmo quem não tiver telas sensíveis ao toque conseguirão se utilizar da nova interface tendo uma experiência satisfatória como a de quem possui? Qual é a linha de aprendizado proporcionado pela nova interface?

A Metro UI, como é conhecida, é uma linguagem de design criada pela Microsoft para ser usada em seu Windows 8. Linguagens de design são arquiteturas para a construção de mecanismos que proporcionam interação entre as máquinas e as pessoas. No núcleo da Metro UI é baseada nos princípios clássicos da escola suíça de design gráfico, conforme a Microsoft, o nome oferecido à linguagem de design representa toda a velocidade, autenticidade e modernidade da tipografia usada, bem como a constante "movimentação" do seu código. Segundo informações vindas da equipe de design da empresa, a tipografia utilizada nessa interface foi inspirada nos símbolos do sistema público de metrô de Londres. A Metro UI foi projetada para ser uma experiência agradável para os usuários, utilizando botões e títulos maiores, além de transições mais suaves para a navegação entre telas. A Microsoft colocou um visual mais limpo, excluindo elementos gráficos desnecessários ao sistema operacional (DAQUINO, 2011).

A interface vem com o intuito de ser personalizada, gerando assim um maior apelo para que o usuário se sinta mais confortável conforme as suas escolhas e não as já definidas pela empresa. O usuário pode ligar pequenos blocos dinâmicos, ou live tiles, conforme suas necessidades como: Notícias do mundo, seus e-mails, pacotes de viagens, mensagens, Skype e etc., dando ao usuário a escolha total do que irá aparecer na sua interface.

O modelo original para o modo desktop, onde havia um botão iniciar, foi alterado. Ao invés de um botão iniciar antigo foi inserido uma chamada para a

interface, ao clicar o usuário é redirecionado para a Metro UI, assim retornando sempre a sua tela personalizada de ações.

1.2 OBJETIVO

A proposta deste trabalho é a criação de um aplicativo voltado para empresas de serviço de entrega de refeições prontas, que integre a Internet ao dispositivo que o usuário está utilizando para realizar seu pedido, recebendo a refeição no endereço solicitado. O aplicativo utilizará o conceito trazido da Metro UI, baseado na integração de dispositivos proposta pelo sistema operacional Windows 8 e assim avaliar a real usabilidade e facilidade de manuseio de sua nova interface.

1.3 PÚBLICO ALVO

Pretende-se para facilitar a comunicação entre cliente e empresa na hora de realizar pedidos de entrega de comida, conhecida por *fast food*, trazendo uma melhoria na agilidade do processo em relação ao tradicional telefone de entrega de cada empresa. Também considera-se como público alvo deste trabalho os profissionais de TI interessados na avaliação da usabilidade da Metro UI.

1.4 JUSTIFICATIVA

O aplicativo tem como objetivo principal trazer uma melhoria no método tradicional de telefones de entrega, agilizando o serviço, já que com poucos cliques o usuário poderá fazer seu pedido e a empresa se responsabiliza pela entrega e também podemos analisar se a interface Metro UI está trazendo melhorias para o usuário na hora de interagir com o Sistema Operacional, tornando cada vez mais fácil a utilização da tecnologia no nosso dia a dia.

1.5 MÉTODO DE DESENVOLVIMENTO

Para o desenvolvimento do aplicativo foi utilizado Visual Studio 2012, utilizando-se da linguagem de programação C# com banco de dados MySQL, utilizando de Web Services em PHP. Também se contará uma analise para mensuração da facilidade ou dificuldade de uso da Metro UI, utilizando a aplicação.

2 AMBIENTE DE DESENVOLVIMENTO

Neste capitulo abordamos o ambiente utilizado para desenvolvimento do aplicativo explicando as ferramentas utilizadas.

2.1 WINDOWS 8

O Windows 8 é a continuação do sistema operacional voltado para usuários comuns da empresa Microsoft, seu nome é a sequência numérica de seu antecessor, Windows 7.

Por conta da popularização de dispositivos como tablets, que vêm para criar uma linha de frente contra os PCs, Notebooks e Netbooks em relação ao mercado de consumidores comuns, assim o Android, sistema aberto da Google e o IOS, sistema portátil da Apple, vem de certa forma tomando uma boa fatia do mercado de sistemas operacionais da Microsoft.

Para a Microsoft a aposta fica em uma espécie de multifaces para o Windows 8, com mercado principal voltado para desktops e notebooks, mas vem com uma segunda faceta: atacar de frente o mercado de tablets e até celulares. As principais evoluções do sistema operacional foram:

· Consumo menor de CPU, com suporte a CPUs relativamente lentas;

· Consumo reduzido de memória;

· Consumo reduzido de energia e bateria;

· Rapidez tanto no uso, quanto no momento de ligar e desligar;

· Possibilidade de rodar a partir de dispositivos como pen drives;

· Modo Restauração onde o estado do sistema operacional pode ser restaurado a partir de um ponto anterior;

· Suporte a novos sensores como luz ambiente, movimento, orientação, acelerômetro, e giroscópio;

· Suporte a novos dispositivos como USB 3.0, WiFi Direct, Bluetooth de baixa potência e "Near Field Communication";

· Boot mais rápido e seguro através de um padrão de BIOS novo (UEFI);

· Suporte à CPUs ARM.

Uma das principais diferenças do Windows 8 sem dúvidas é a interface Metro, que vêm para agregar um pouco do lado portátil do sistema, suas características principais são:

· Uma interface com usuário completamente nova;

· Um novo modelo de distribuição;

· Um novo conjunto de APIs;

· A computação na nuvem muito mais presente.

A Metro vem para trabalhar de maneira dupla, de um lado mantendo a linha criada nos dispositivos móveis com todas as funcionalidades para dispositivos sensíveis ao toque, e claro a antiga maneira de trabalhar com WinForms e WPF, estes modelos continuam totalmente disponíveis. Softwares como o Visual Studio, Photoshop, Word, Excel, entre outros continuarão a usar o modelo antigos de desktop (SANT'ANNA, 2011).

O layout da Metro é muito parecido do que já vou visto no Windows Phone 7. Utiliza-se de um modelo limpo, fluído, cores básicas, tipografia simples e é otimizada para uso com interface a toque.

A interface em si, sem o modo desktop não suporta janela, uma característica essencial de todas as outras versões do Windows. Todas as aplicações são em tela cheia nesse novo modelo, como visto em seus concorrentes móveis (Android e IOS), assim como todas as telas de interface das aplicações, sem mais janelas sobre janelas, caixas de diálogos, mensagens e tudo que acabava poluindo o espaço visível da tela.

A interface Metro deve ser rápida e fluida, como foi proposto pela empresa, ela acrescenta certas responsabilidades em relação aos aplicativos, eles devem sempre responder, caso algum trave, atrapalhara o funcionamento da interface, para ajudar nesta parte a Microsoft utiliza uma solução já implementada em outras versões do seu sistema: multi-thread. O uso de multi-threads é algo ainda extremamente complexo e difícil de fazer direito. O Windows 8 traz então apenas APIs assíncronas – sem suprir as variedades síncronas, de forma a evitar o seu uso. Para facilitar o uso, as linguagens suportam um mecanismo de "co-rotinas", que é uma solução simples e eficaz para o problema (MIKKO, 2011).

O sistema operacional também controla todos os aplicativos e encerra o processamento de aplicativos que pararam de responder, fazendo com que o sistema mantenha-se sempre fluido.

Uma das atualizações é um modelo baseado em sua Windows Store, a loja de aplicativos. Os arquivos são criados em um único arquivo, onde o aplicativo encontrará tudo que ele precisa, as restrições que ele terá também, ao usar o sistema, isto evita que o aplicativo não acesse mais do que o sistema permite para ele, aumentando a segurança e a integridade do sistema. O aplicativo irá evitar de acumular lixo na instalação e ao longo do uso.

Todos os aplicativos têm um documento que declara os recursos necessários e o sistema de execução valida as chamadas para respeitar a lista de recursos declarados.

Utilizando a DRM ("Digital Rights Management", em português "Gestão de Direitos Digitais") possibilita aos que querem vender seus aplicativos apresentem uma garantia que seus aplicativos não serão pirateados (ou serão pouco pirateados). Quebrando o ciclo de que os "aplicativos são pirateados porque o preço é muito caro e são caros porque são pirateados demais", em um modelo já provado com sucesso pela Apple.

A desinstalação no Windows 8 será completo, os aplicativos simplesmente não podem deixar pedaços espalhados, o que contribui bastante para a estabilidade do sistema operacional.

As principais diferenças do Windows 8 está nas APIs. A antinga API Win32, existente desde a primeira versão do Windows não é mais suportada nos aplicativos Metro. Ela foi substituída por uma API chamada WinRT, nesta API é orientada a objeto e feita em C++, que roda diretamente o runtime da aplicação, ela pode ser chamada a partir de código gerenciado em C#, VB.NET e JavaScript. Muitas APIs gerenciadas estão disponíveis, mas alguma não ou foram substituídas por chamadas nativas "WinRT" de forma a não haver sobreposição, então o ambiente de desenvolvimento mesmo para C# e VB.NET tem pequenas diferenças.

Há também a possibilidade de desenvolver aplicativos utilizando HTML/JavaScript/CSS, inclusive jQuery e HTML5.

O ciclo da aplicação é feito pelo sistema e não pelo usuário. Na Metro o usuário abre os aplicativos, mas quem os fecha é o sistema. Inicialmente nenhum aplicativo pode rodar em segundo plano, mas existem APIs para rodar coisas em intervalos pré-determinados ou como resposta a notificações. Isso traz uma grande economia de bateria, grande trunfo para utilizar-se o sistema em dispositivos móveis.

2.2 MICROSOFT .NET

A Microsoft .NET, ou .NET somente, lê-se em inglês dot Net, é a plataforma da Microsoft única para desenvolvimento e execução de sistemas e aplicações. Qualquer código gerado para .NET pode ser executado em qualquer dispositivo que possua um framework da plataforma. Assim o programador não fica restrito a escrever um código para um sistema ou dispositivo apenas, ele escreve para a plataforma .NET que irá executar em todos os dispositivos e sistemas que rodem ela.

Sendo executada sobre uma CLR (Common Language Runtime, um Ambiente de Execução Independente de Linguagem) a plataforma .NET interage com um framework (Conjunto de Bibliotecas Unificadas), com esta CLR a plataforma se torna capaz de executar linguagens diferentes de programação, interagindo entre como se fossem uma única linguagem (TROELSEN, 2010).

Utilizando o conceito de uma das vertentes utilizadas na tecnologia Java, o JIT(Just In Time Compiler), todos os programas escritos serão duplo-compilados, ou seja compilados duas vezes, uma vez no momento da distribuição (criando assim um código que é conhecido como "bytecodes") e outra no momento da execução.

Toda vez que o código fonte escrito é compilado é gerado um código intermediário na linguagem MSIL (Microsoft Intermediate Language) (TROELSEN, 2010).

O código fonte compilado na MSIL gera um arquivo na linguagem de baixo nível Assembly, conforme o tipo do projeto (EXE, DLL, ASPX e ASMX).

Utilizando-se a MSIL como intermediária cria a possibilidade haver a famosa "engenharia reversa", que basicamente é: a partir de um código compilado, recuperar o código original. Por conta desta situação existem ferramentas que dificultam a realização da engenharia reversa trocando nomes de variáveis, métodos, interfaces e etc no código da MSIL (TROELSEN, 2010).

2.3 LINGUAGEM DE PROGRAMAÇÃO C#

A linguagem de programação C# ou C Sharp é uma linguagem de programação orientada a objetos, fortemente tipada, desenvolvida pela Microsoft como parte da plataforma .NET. Baseada principalmente em C++ sua sintaxe orientada a objetos, porém engloba muitas influências de outras linguagens de programação, como por exemplo Object Pascal e Java (TROELSEN, 2010).

O C# entra no conjunto de linguagens que tem plataforma .NET com principais características de ser simples, robusto, orientada a objetos, fortemente tipada e altamente escalável com a finalidade de permitir a portabilidade para diferentes dispositivos e hardwares desde que rodem a plataforma .NET (TROELSEN, 2010).

O avanço da tecnologia e dos dispositivos eletrônicos trouxe novos patamares de exigências que as novas versões de componentes compartilhados eram

conflitantes com o software antigo. Então a necessidade de que o software fosse desenvolvido de um jeito que se tornasse acessível para qualquer dispositivo se tornou clara neste mercado. Com isso a Microsoft lançou sua plataforma .NET, vista anteriormente, com a linguagem de programação C# totalmente preparada para esta realidade.

Quando a plataforma .NET começou a ser desenvolvida as bibliotecas foram escritas inicialmente em Simple Managed C (SMC), uma linguagem que tinha um seu próprio compilador. Em 1999, foi escolhida pela Microsoft uma equipe de desenvolvimento formada por Anders Hejlsberg, para à criação da linguagem Cool. O .NET foi apresentado já em 2000 na Professional Developers Conference (PDC), e a linguagem que antes era conhecida como Cool foi apresentada com um novo nome, C#.

Anders é considerado o principal criador da linguagem, claro que com o auxílio de sua equipe de desenvolvimento. Foi o principal arquiteto de compiladores da Borland, e entre seus trabalhos mais famosos estão o Turbo Pascal e o Delphi.

O C# também têm alguns compiladores em outras plataformas, ainda em desenvolvimento, como por exemplo a Mono, implementação open source da Novell, o dotGNU e o Portable.NET, implementações da Free Software Foundation, e o BDS 2008, implementação da CodeGear (TROELSEN, 2010).

2.4 MySQL

O MySQL é um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL (Linguagem de Consulta Estruturada, do inglês Structured Query Language) como interface (NEVES, 2005).

Desenvolvido por David Axmark, Allan Larsson e Michael "Monty" Widenius. As características principais do MySQL são:

Portabilidade (suporta praticamente qualquer plataforma atual); Compatibilidade (existem drivers ODBC, JDBC e .NET e módulos de interface para diversas linguagens de programação, como Delphi, Java, C/C++, C#, Visual Basic, Python, Perl, PHP, ASP e Ruby);

Excelente desempenho e estabilidade; Pouco exigente quanto a recursos de novos hardware; Facilidade no manuseio;

É um Software Livre com base na GPL (entretanto, se o programa que acessar o Mysql não for GPL, uma licença comercial deverá ser adquirida) ;

Contempla a utilização de vários Storage Engines como MyISAM, InnoDB, Falcon, BDB, Archive, Federated, CSV, Solid; Suporta controle transacional;

Suporta Triggers; Suporta Cursors (Non-Scrollable e Non-Updatable); Suporta Stored Procedures e Functions;

Replicação facilmente configurável; Interfaces gráficas (MySQL Toolkit) de fácil utilização cedidos pela MySQL Inc (NEVES, 2005).

2.5 WINDOWS STORE

A Windows Store é a loja de aplicativos da Microsoft, ela vem com a fórmula de sucesso estabelecida pela Apple e pelo Google Android, com a App Store e o Android Marketplace, e traz para as plataformas de desktops algo semelhante ao que os usuários do Windows Phone já conhecem com o Marketplace (DE SOUZA, 2013).

O principal objetivo da Windows Store é oferecer aos usuários do Windows 8 uma forma simples e objetiva de encontrar um novo software para o seu sistema. O usuário poderá encontrar aplicativos desenvolvido pela Microsoft e aplicativos de terceiros e/ou de desenvolvedores independentes, com a maioria deles adaptados para a interface Metro. Aplicativos já consagrados, como o jogo "Cut the Rope" e a versão digital do jornal USA Today estão disponíveis para download na Windwos Store. Atualmente (até Julho de 2013) a loja já recebeu mais de 100 mil aplicativos.

A loja oferece versões de aplicativos tanto para processadores no padrão ARM quanto no padrão x86, identificando automaticamente qual é a sua versão, e fazendo o download da versão correta do aplicativo.

Os aplicativos são agrupados na loja por tipo, tornando a navegação mais prática e intuitiva. A Windows Store também conta com um elemento relacionado à nuvem, onde os usuários fazem login em qualquer computador, com Windows 8 instalado, tendo acesso a todos os seus aplicativos e configurações instaladas na máquina principal (KEENE, 2012).

2.6 LINGUAGEM DE PROGRAMAÇÃO PHP

O PHP é um acrônimo recursivo para "PHP: Hypertext Preprocessor", originalmente Personal Home Page, é uma linguagem interpretada livre, ela era usada inicialmente apenas para o desenvolvimento de aplicações presentes e atuantes no lado do servidor, capazes de gerar conteúdo dinâmico.

No PHP o código é interpretado no lado do servidor, que também gera a página web a ser visualizada no lado do cliente.

Foi criada por Rasmus Lerdorf em 1995, o PHP tem a produção de sua implementação principal — referência formal da linguagem, mantida por uma organização chamada The PHP Group. O PHP é software livre, licenciado sob a PHP License, uma licença incompatível com a GNU General Public License (GPL) devido a restrições no uso do termo PHP(OLSON, 2013).

2.6 SLIM FRAMEWORK

Slim Framework é um framework desenvolvido em PHP que ajuda você a programar rapidamente poderosas aplicações web e APIs, baseadas ou não em REST.

3 MODELAGEM

Neste capitulo é realizada a modelagem de dados necessária para a programação.

3.1 REQUISITOS

- Manter empresa
- Emitir relatório de pedidos feitos

- Manter clientes
- Realizar pedido

- Manter catalogo da empresa
- Exibir catalogo

- Manter cidade
- Emitir relatório de cidade

- Manter estado
- Emitir relatório de estado

3.1.2 ATORES

- Empresa
- Cliente
- Administrador

3.2 WBS - Work Breakdown Structure

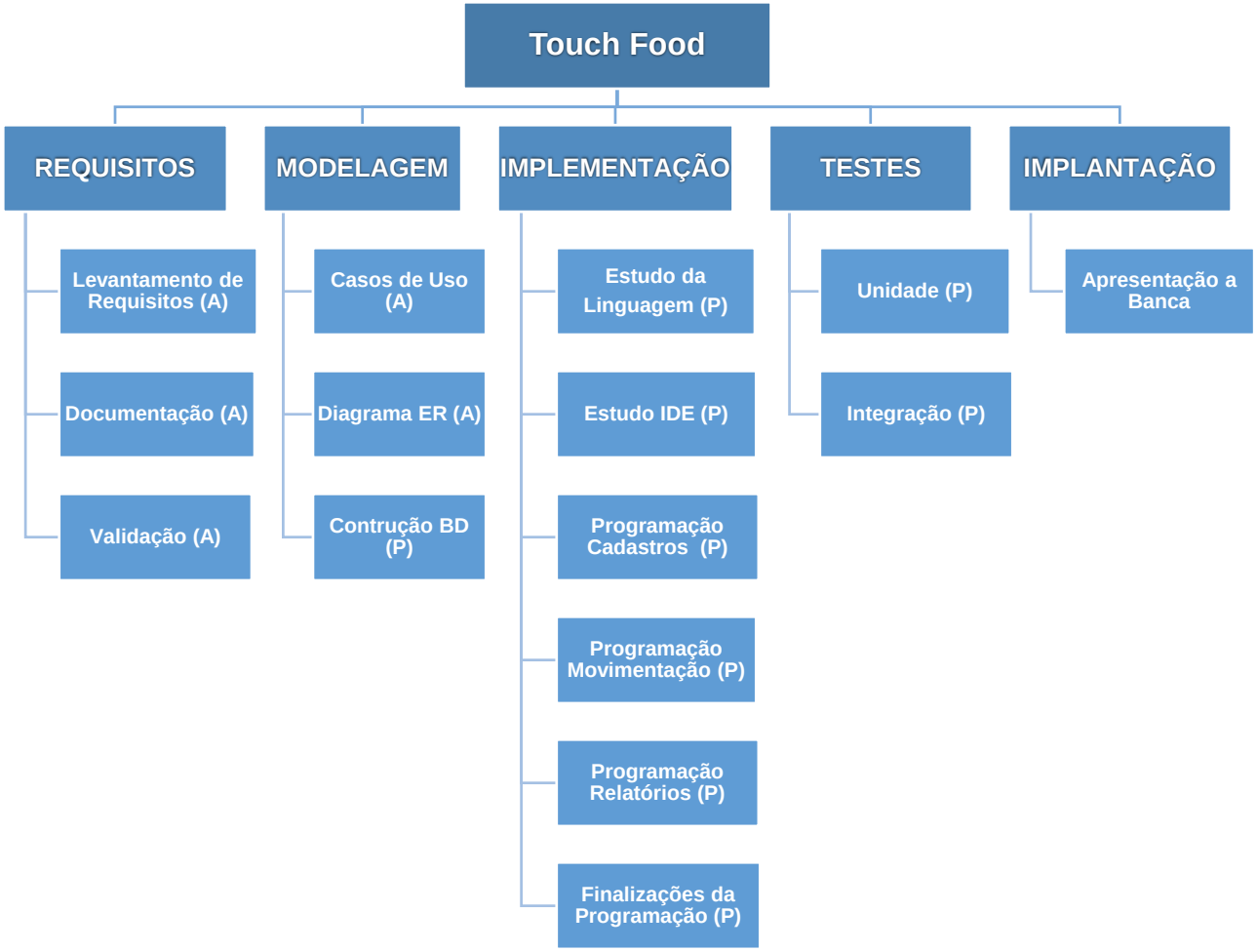

Figura 1: WBS Touch Food

3.3 CASOS DE USO

Diagrama de Casos de Uso

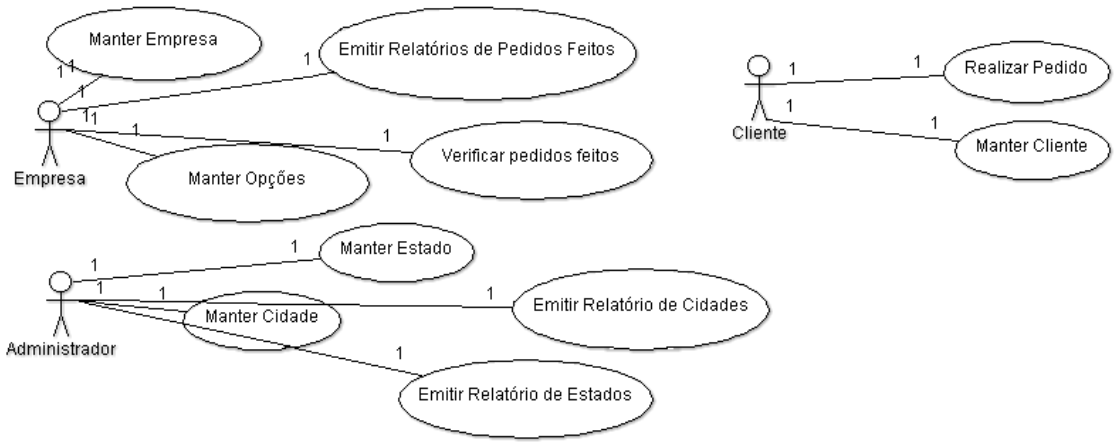

Figura 2: Casos de Uso

Casos de Uso: UC01 - Manter Empresa

1. Finalidade / Objetivo
   - Permite a empresa inserir, procurar, excluir e alterar um cadastro de empresa.

2. Atores
   - Empresa.

3. Pré-condições
   - A empresa deve acessar a tela inicial do sistema.

4. Evento inicial
   - A empresa escolhe no menu da tela inicial do sistema a opção Realizar Cadastro.

5. Fluxo principal

5.1 Inserir

a. O sistema exibe um formulário para inserir: razão social, login, senha, cnpj,telefone e email;
b. A empresa fornece as informações solicitadas; (A1)(A2)

Fluxos Alternativos

A1 – Opção voltar

a. A empresa clica no botão voltar, cancelando a operação;

b. O caso de uso é encerrado.

A2 – Confirma a operação
a. A empresa confirma a operação; (E1)
b. O caso de uso é encerrado.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente
a. O sistema verifica que já existe uma empresa cadastrado com o cnpj informado e emite uma mensagem de "CNPJ já cadastrado";

5.2 Editar
a. O sistema exibe um formulário de cadastro efetuado; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
a. A empresa cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirma a operação
a. O administrador confirma a operação; (E1)(E2)
b. O sistema atualiza o banco de dados e exibe uma mensagem de aviso informando "Empresa alterada com sucesso";
c. O caso de uso é encerrado.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente
a. O sistema identifica que o campo confirmar senha não foi preenchido de acordo com o campo senha e exibe a mensagem "Falha na confirmação de senha";

E2 – Campo obrigatório não preenchido
a. O sistema identifica se ao menos um dos campos não foi preenchido;
b. O sistema exibe uma mensagem de aviso informando que "Os dados não foram incluídos. Informações restantes a serem preenchidas";
c. O sistema aguarda preenchimento total.

5.4 Excluir
a. A empresa deseja excluir seu cadastro; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
a. A empresa cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirmação de Exclusão
a. A empresa confirma a exclusão;
b. O sistema exclui o cadastro do banco de dados e exibe uma mensagem de aviso informando que "Empresa excluída com sucesso";
c. O caso de uso é encerrado.

Fluxos de Exceção
Não há.

6. Pós-condições
- Manter cadastro de empresas atualizado.

7. Casos de testes
- Cadastrar empresa;
- Editar dados de empresa;
- Excluir cadastro de empresa;
- Cancelar operação de cadastro;
- Cancelar operação de edição;
- Cancelar operação de exclusão;
- Verificar o preenchimento de todos os campos;
- Verificar se o campo senha e confirmar senha foram preenchidos corretamente;
- Verificar se a empresa já está cadastrada;

Casos de Uso: UC02 - Manter Cliente

1. Finalidade / Objetivo
- Permite ao cliente inserir, procurar, excluir e alterar cadastro de cliente.

2. Atores
- Cliente.

3. Pré-condições
- O cliente deve estar na tela inicial do sistema.

4. Evento inicial
- O cliente escolhe na tela inicial do sistema a opção Realizar Cadastro.

5. Fluxo principal

5.1 Inserir

a. O sistema exibe um formulário para inserir: nome, rg, cpf, login, senha, rua, cidade, estado, número e telefone;
b. O cliente fornece as informações solicitadas; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
a. O cliente cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirma a operação
a. O cliente confirma a operação;
b. O sistema atualiza o banco de dados e exibe a mensagem "Cliente incluído com sucesso"; (E1)(E2)
c. O caso de uso é encerrado.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido
a. O sistema identifica se ao menos um dos campos não foi preenchido;
b. O sistema exibe uma mensagem de aviso informando "Os dados não foram incluídos. Informações restantes a serem preenchidas";
c. O sistema aguarda preenchimento total

E2 – Cliente já cadastrado
a. O sistema verifica que já existe um cliente cadastrado com o cpf informado e emite uma mensagem "CPF já cadastrado";

5.2 Editar

a. O sistema exibe um formulário de cadastro efetuado; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
a. O cliente cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirma a operação
a. O cliente confirma a operação; (E1)
b. O sistema atualiza o banco de dados e exibe uma mensagem de aviso informando "Cliente alterado com sucesso";
c. O caso de uso é encerrado.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido
a. O sistema identifica se ao menos um dos campos não foi preenchido;
b. O sistema exibe uma mensagem de aviso informando "Os dados não foram incluídos. Informações restantes a serem preenchidas";
c. O sistema aguarda o preenchimento total.

5.4 Excluir
a. O cliente faz escolhe excluir seu cadastro; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
a. O cliente cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirmação de Exclusão
a. O administrador confirma a exclusão;
b. O sistema exclui o cadastro do banco de dados exibe uma mensagem de aviso informando que "Cliente excluído com sucesso";
c. O caso de uso é encerrado.

Fluxos de Exceção
Não há.

6. Pós-condições
- Manter cadastro de clientes atualizado.

7. Casos de testes
- Cadastrar clientes;

- Editar cadastro de clientes;
- Excluir cadastro de clientes;
- Cancelar operação de cadastro;
- Cancelar operação de edição;
- Cancelar operação de exclusão;
- Verificar o preenchimento de todos os campos;
- Verificar se o cliente já está cadastrado;

Caso de Uso: UC03 - Manter Opções

1. Finalidade / Objetivo
- Permite a empresa inserir, procurar, excluir e alterar cadastro de opções em seu cardápio.

2. Atores
- Empresa.

3. Pré-condições
- A empresa deve realizar o login e estar na tela inicial do sistema.

4. Evento inicial
- A Empresa escolhe na tela inicial do sistema a opção Cardápio/Cadastro de opções.

5. Fluxo principal

5.1 Inserir
a. O sistema exibe um formulário para inserir: nome, imagem, descrição e preço;
b. A empresa fornece as informações solicitadas; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
a. A empresa cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirma a operação
a. A empresa confirma a operação;
b. O sistema atualiza o banco de dados e exibe a mensagem "Opção incluída com sucesso"; (E1)(E2)
c. O caso de uso é encerrado.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido
a. O sistema identifica se ao menos um dos campos não foi preenchido;
b. O sistema exibe uma mensagem de aviso informando "Os dados não foram incluídos. Informações restantes a serem preenchidas";
c. O sistema aguarda preenchimento total.

5.2 Procurar
a. A empresa clica no campo "Procurar" e informa a opção a ser buscada; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
a. A empresa cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirma a operação
a. A empresa confirma a operação; (E1)
b. O sistema alimenta todos os campos. (A1)

Fluxos de Exceção

E1 – Não ter opções cadastradas
a. O sistema realiza a busca e verifica que não existem opções cadastradas com os dados informados, exibe uma mensagem de aviso informando "Opção não encontrada";
b. O caso de uso é encerrado.

5.3 Editar
a. O sistema exibe um formulário de cadastro efetuado; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
a. A empresa cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirma a operação
a. O administrador confirma a operação; (E1)
b. O sistema atualiza o banco de dados e exibe uma mensagem de aviso informando "Opção alterada com sucesso";
c. O caso de uso é encerrado

Fluxos de Exceção

E1 – Campo obrigatório não preenchido
a. O sistema identifica se ao menos um dos campos não foi preenchido;
b. O sistema exibe uma mensagem de aviso informando "Os dados não foram incluídos. Informações restantes a serem preenchidas";

5.4 Excluir
a. A empresa faz a busca escolhe a opção a ser excluído; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
a. A empresa cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirmação de Exclusão

a. A empresa confirma a exclusão;
b. O sistema exclui o cadastro do banco de dados exibe uma mensagem de aviso informando que "Opção excluída com sucesso";
c. O caso de uso é encerrado.

Fluxos de Exceção

Não há.

6. Pós-condições

- Manter cadastro de opções atualizado.

7. Casos de testes

- Cadastrar cidades;
- Editar cadastro de opções;
- Excluir cadastro de opções;
- Buscar opções por nome;
- Cancelar operação de cadastro;
- Cancelar operação de edição;
- Cancelar operação de busca;
- Cancelar operação de exclusão;
- Verificar o preenchimento de todos os campos;
- Consultar todas as opções cadastradas;

Caso de Uso: UC05 – Verificar Pedidos Feitos

1. Finalidade / Objetivo
- Permite a empresa procurar e marcar como entregue pedidos feitos por cliente.

2. Atores
- Empresa.

3. Pré-condições
- A empresa deve realizar o login e estar na tela inicial do sistema.

4. Evento inicial
- A empresa escolhe na tela inicial do sistema a opção Pedidos.

5. Fluxo principal

5.1 Procurar
a. O sistema exibe um formulário listando os pedidos realizados;
b. A empresa escolhe qual pedido ver; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
a. A empresa cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirma a operação
a. A empresa confirma a operação;
b. O sistema exibe o nome do cliente e o pedido do cliente;
c. O caso de uso é encerrado.

5.2 Entregar
a. A empresa clica no campo "Entregar" e informa o tempo estimado para a entrega; (A1)

Fluxos Alternativos

A1 – Cancela a operação
a. A empresa cancela a operação, uma mensagem é exibida para o cliente informando o cancelamento;
b. O caso de uso é encerrado.

6. Pós-condições
- Manter pedidos atualizados.

7. Casos de testes

- Verificar pedidos;
- Buscar pedidos;
- Cancelar pedido;
- Cancelar operação de busca;
- Verificar pedido entregue;

Caso de Uso: UC06 – Realizar Pedido

1. Finalidade / Objetivo
- Permite ao cliente realizar e conferir um pedido.

2. Atores
- Cliente.

3. Pré-condições
- O cliente deve realizar o login e estar na tela inicial do sistema.

4. Evento inicial
- O cliente escolhe na tela inicial do sistema a opção Empresa, ele acessara um catálogo com as opções da empresa.

5. Fluxo principal

5.1 Pedido
a. O sistema exibe opções de pedido, onde o cliente deve escolher uma opção e escolher o endereço de entrega;
b. O cliente fornece as informações solicitadas; (A1)(A2)(A3)

Fluxos Alternativos

A1 – Cancela a operação
a. O cliente cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirma a operação
a. O cliente confirma a operação;
b. O sistema atualiza o banco de dados e exibe a mensagem "Pedido realizado com sucesso";
c. O exibe o acompanhamento do pedido;

A3 – Cancelar pedido
a. O cliente apaga cancela pedido, uma mensagem é enviada a empresa; (E1)

Fluxos de Exceção

E1 – Pedido já saiu para entrega
a. O sistema identifica se a empresa já marcou como saída para entrega;
b. O sistema exibe uma mensagem de aviso informando "O pedido não pode ser cancelado, pois já saiu para entrega";
c. O caso de uso é encerrado.

5.2 Conferir

a. O cliente clica no campo "Acompanhar" e seleciona o pedido feito; (A1)(A2) (E1)

Fluxos Alternativos

A1 – Cancela a operação

a. O cliente cancela a operação;
b. O caso de uso é encerrado.

A2 – Confirma a operação

a. O cliente confirma a operação;
b. O sistema alimenta todos os campos. (A1)

Fluxos de Exceção

E1 – Não ter pedidos cadastrados

a. O sistema realiza a busca e verifica que não existem pedidos cadastrados, exibe uma mensagem de aviso informando "Não há pedidos";
b. O caso de uso é encerrado.

6. Pós-condições

- Manter pedidos atualizados.

7. Casos de testes

- Realizar pedido;
- Buscar pedidos;
- Cancelar operação de pedido;
- Cancelar operação de busca;
- Cancelar operação de cancela;
- Consultar os pedidos realizado;

Caso de Uso: UC07 – Emitir Relatório de Pedidos Feitos

1. Finalidade / Objetivo
   - Permite a empresa obter um relatório de todos os pedidos realizados pelo sistema.

2. Atores
   - Empresa.

3. Pré-condições
   - A empresa deve realizar o login e estar na tela inicial do sistema.

4. Evento inicial
   - A empresa escolhe na tela inicial do sistema a opção Relatórios/Pedidos.
5. Fluxo principal

   - O sistema exibe um relatório com as informações: pedidos, opção pedida, data, horário;

   Fluxos Alternativos:
   Não há.

   Fluxos de Exceção
   Não há.

6. Pós-condições
   - Manter pedidos atualizados.

7. Casos de testes
   - Atualizar relatório;
   - Imprimir relatório;
   - Visualizar impressão de relatório;
   - Ajustar layout da página para impressão;
   - Exportar para PDF;
   - Verificar o preenchimento de todos os campos;

Caso de Uso: UC08 – Exibir Catálogo de Opções

1. Finalidade / Objetivo
   - Permite ao cliente obter um catálogo com todos os opções da empresa.

2. Atores
   - Cliente.

3. Pré-condições
   - O cliente deve realizar o login e estar na tela inicial do sistema.

4. Evento inicial
   - O cliente escolhe na tela inicial do sistema um empresa, automaticamente já irá aparecer um catálogo atualizado de opções.
5. Fluxo principal

   - O sistema exibe uma tela com as informações: opções e valores;

   Fluxos Alternativos:
   Não há.

   Fluxos de Exceção
   Não há.

6. Pós-condições
   - Manter opções atualizados.

7. Casos de testes
   - Acessar catalogo;
   - Verificar se as opções estão sendo exibidas de maneira adequada;

Caso de Uso: UC09 – Manter Cidade

1. Finalidade / Objetivo
- Permite ao administrador inserir, procurar, excluir e alterar um cadastro de cidade.

2. Atores
- Administrador.

3. Pré-condições
- O administrador deve acessar a tela inicial do sistema.

4. Evento inicial
- O administrador escolhe no menu da tela inicial do sistema a opção Cadastro/Cidade.

5. Fluxo principal

5.1 Inserir

c. O sistema exibe um formulário para inserir: nome e estado;
d. O administrador fornece as informações solicitadas; (A1)(A2)

Fluxos Alternativos

A1 – Opção voltar
c. O administrador clica no botão voltar, cancelando a operação;
d. O caso de uso é encerrado.

A2 – Confirma a operação
c. O administrador confirma a operação; (E1)
d. O caso de uso é encerrado.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente
b. O sistema verifica que já existe uma cidade cadastrada com o nome e estado informado e emite uma mensagem de "Cidade já cadastrada";

5.2 Editar

b. O sistema exibe um formulário de cadastro efetuado; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
c. O administrador cancela a operação;
d. O caso de uso é encerrado.

A2 – Confirma a operação

d. O administrador confirma a operação; (E1)
e. O sistema atualiza o banco de dados e exibe uma mensagem de aviso informando "Cidade alterada com sucesso";
f. O caso de uso é encerrado.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido
d. O sistema identifica se ao menos um dos campos não foi preenchido;
e. O sistema exibe uma mensagem de aviso informando que "Os dados não foram incluídos. Informações restantes a serem preenchidas";
f. O sistema aguarda preenchimento total.

5.4 Excluir
b. O administrador deseja excluir seu cadastro; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
d. O administrador cancela a operação;
e. O caso de uso é encerrado.

A2 – Confirmação de Exclusão
c. O administrador confirma a exclusão;
d. O sistema exclui o cadastro do banco de dados e exibe uma mensagem de aviso informando que "Cidade excluída com sucesso";
f. O caso de uso é encerrado.

Fluxos de Exceção
Não há.

6. Pós-condições
- Manter cadastro de cidades atualizado.

7. Casos de testes
- Cadastrar cidades;
- Editar dados de cidade;
- Excluir cadastro de cidade;
- Cancelar operação de cadastro;
- Cancelar operação de edição;
- Cancelar operação de exclusão;
- Verificar o preenchimento de todos os campos;
- Verificar se a cidade já está cadastrada;

Caso de Uso: UC10 – Manter Estado

1. Finalidade / Objetivo
- Permite ao administrador inserir, procurar, excluir e alterar um cadastro de estado.

2. Atores
- Administrador.

3. Pré-condições
- O administrador deve acessar a tela inicial do sistema.

4. Evento inicial
- O administrador escolhe no menu da tela inicial do sistema a opção Cadastro/Estado.

5. Fluxo principal

5.1 Inserir

e. O sistema exibe um formulário para inserir: nome e sigla;
f. O administrador fornece as informações solicitadas; (A1)(A2)

Fluxos Alternativos

A1 – Opção voltar
e. O administrador clica no botão voltar, cancelando a operação;
f. O caso de uso é encerrado.

A2 – Confirma a operação
e. O administrador confirma a operação; (E1)
f. O caso de uso é encerrado.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente
c. O sistema verifica que já existe um estado cadastrado com o estado informado e emite uma mensagem de "Estado já cadastrada";

5.2 Editar

c. O sistema exibe um formulário de cadastro efetuado; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação
e. O administrador cancela a operação;
f. O caso de uso é encerrado.

A2 – Confirma a operação

g. O administrador confirma a operação; (E1)
h. O sistema atualiza o banco de dados e exibe uma mensagem de aviso informando "Estado alterado com sucesso";
i. O caso de uso é encerrado.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido

g. O sistema identifica se ao menos um dos campos não foi preenchido;
h. O sistema exibe uma mensagem de aviso informando que "Os dados não foram incluídos. Informações restantes a serem preenchidas";
i. O sistema aguarda preenchimento total.

5.4 Excluir

c. O administrador deseja excluir seu cadastro; (A1)(A2)

Fluxos Alternativos

A1 – Cancela a operação

g. O administrador cancela a operação;
h. O caso de uso é encerrado.

A2 – Confirmação de Exclusão

e. O administrador confirma a exclusão;
f. O sistema exclui o cadastro do banco de dados e exibe uma mensagem de aviso informando que "Estado excluído com sucesso";
i. O caso de uso é encerrado.

Fluxos de Exceção

Não há.

6. Pós-condições

- Manter cadastro de cidades atualizado.

7. Casos de testes

- Cadastrar estados;
- Editar dados de estado;
- Excluir cadastro de estado;
- Cancelar operação de cadastro;
- Cancelar operação de edição;
- Cancelar operação de exclusão;
- Verificar o preenchimento de todos os campos;
- Verificar se a estado já está cadastrado;

3.4 SEQUENCIAMENTO DE ATIVIDADES

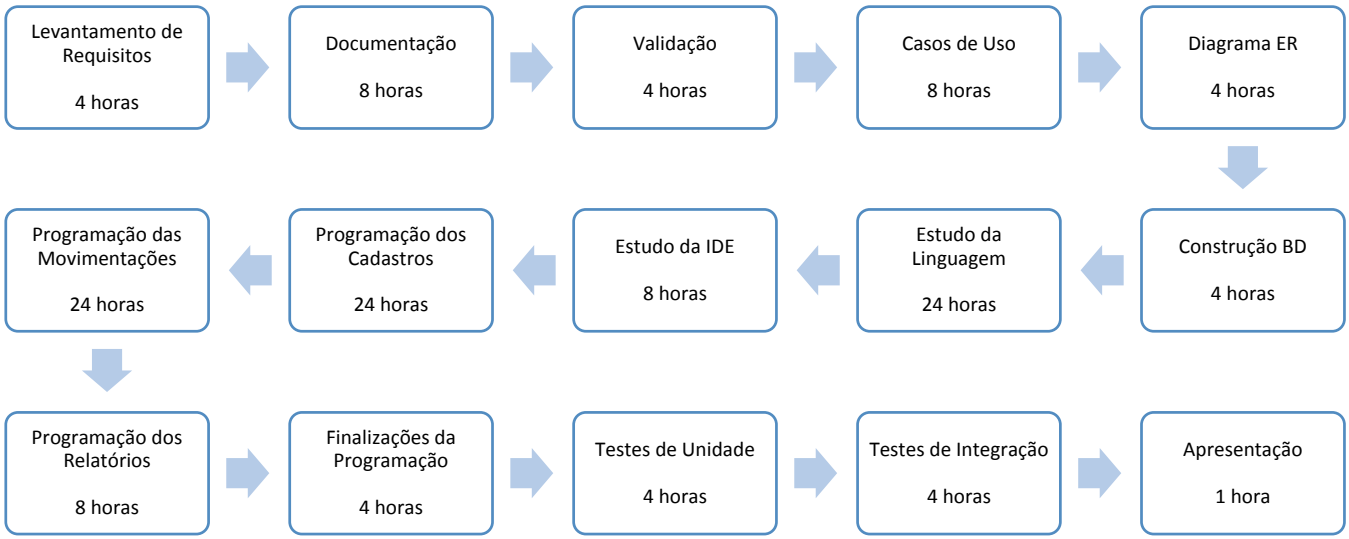

Figura 3: Sequenciamento de Atividades

3.5 DER

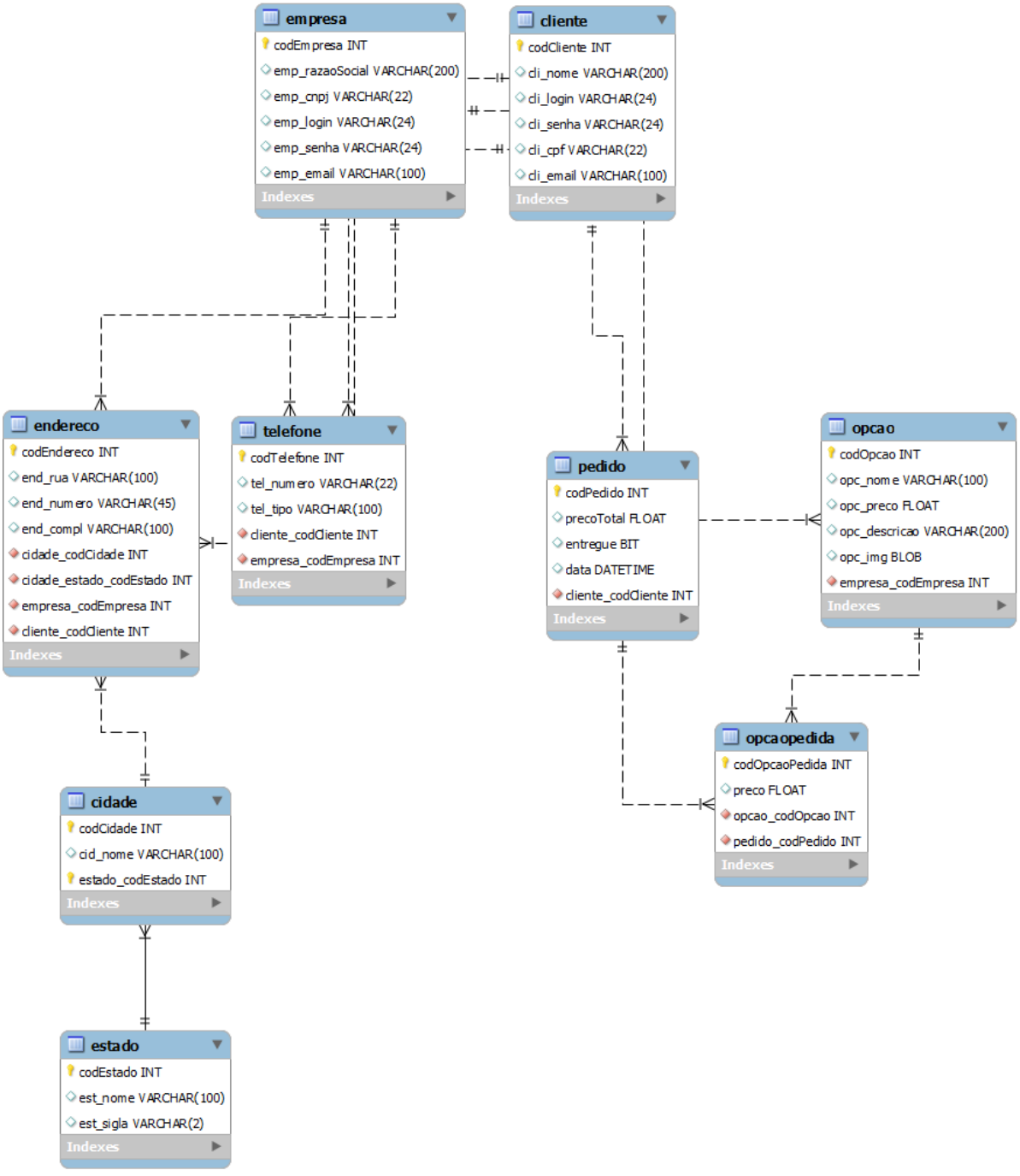

Figura 4: DER

3.6 PROTÓTIPOS DAS TELAS

Login

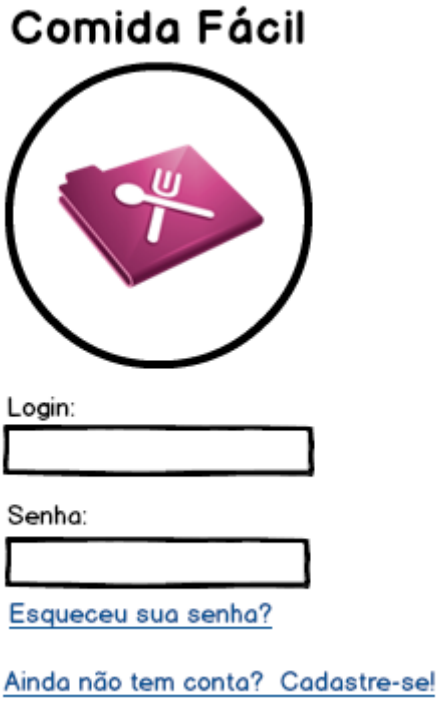

Figura 5: Protótipo de Tela de Login

Manter Clientes

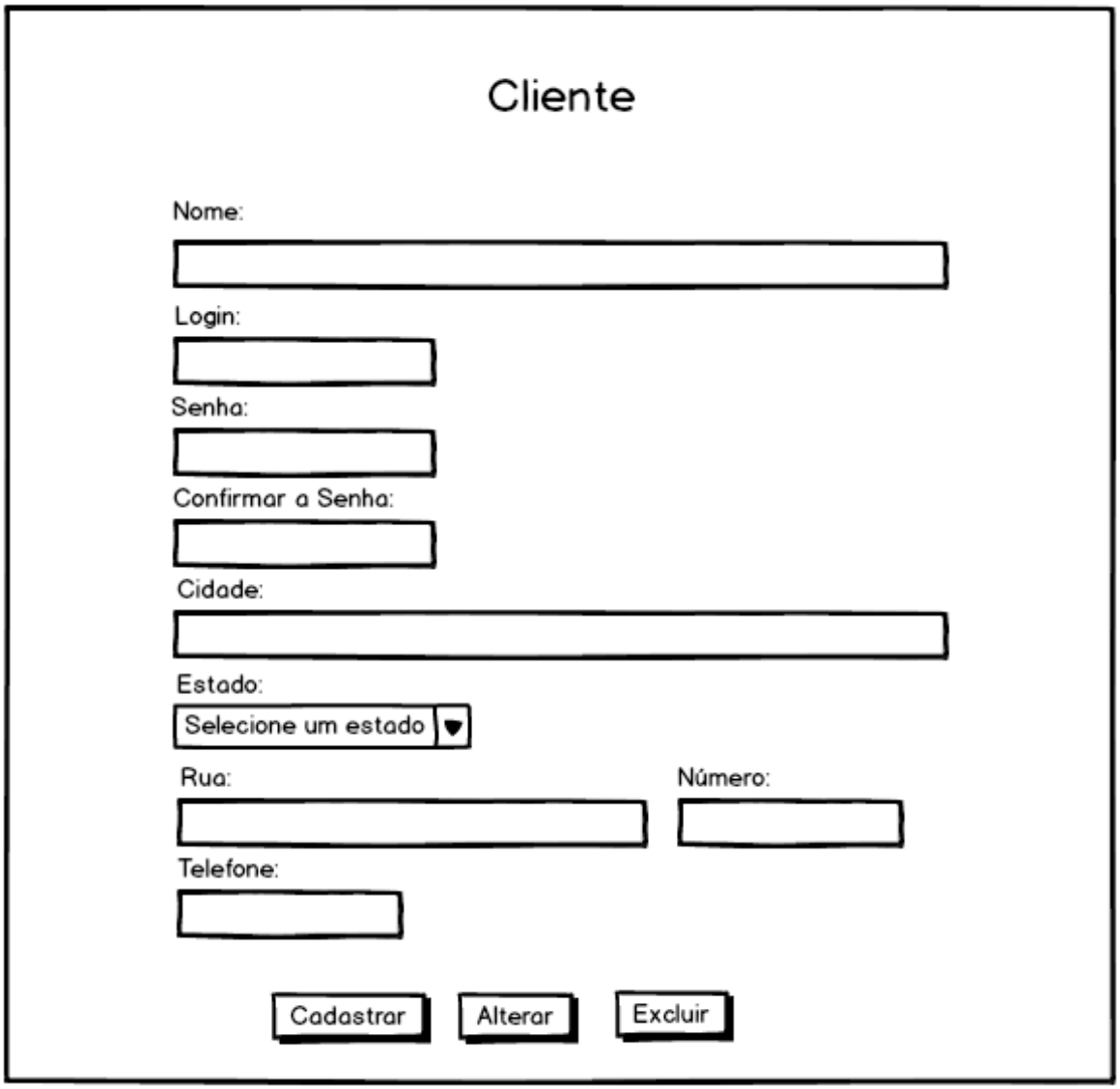

Figura 6: Protótipo de Tela de cadastro de Clientes

Manter Empresas

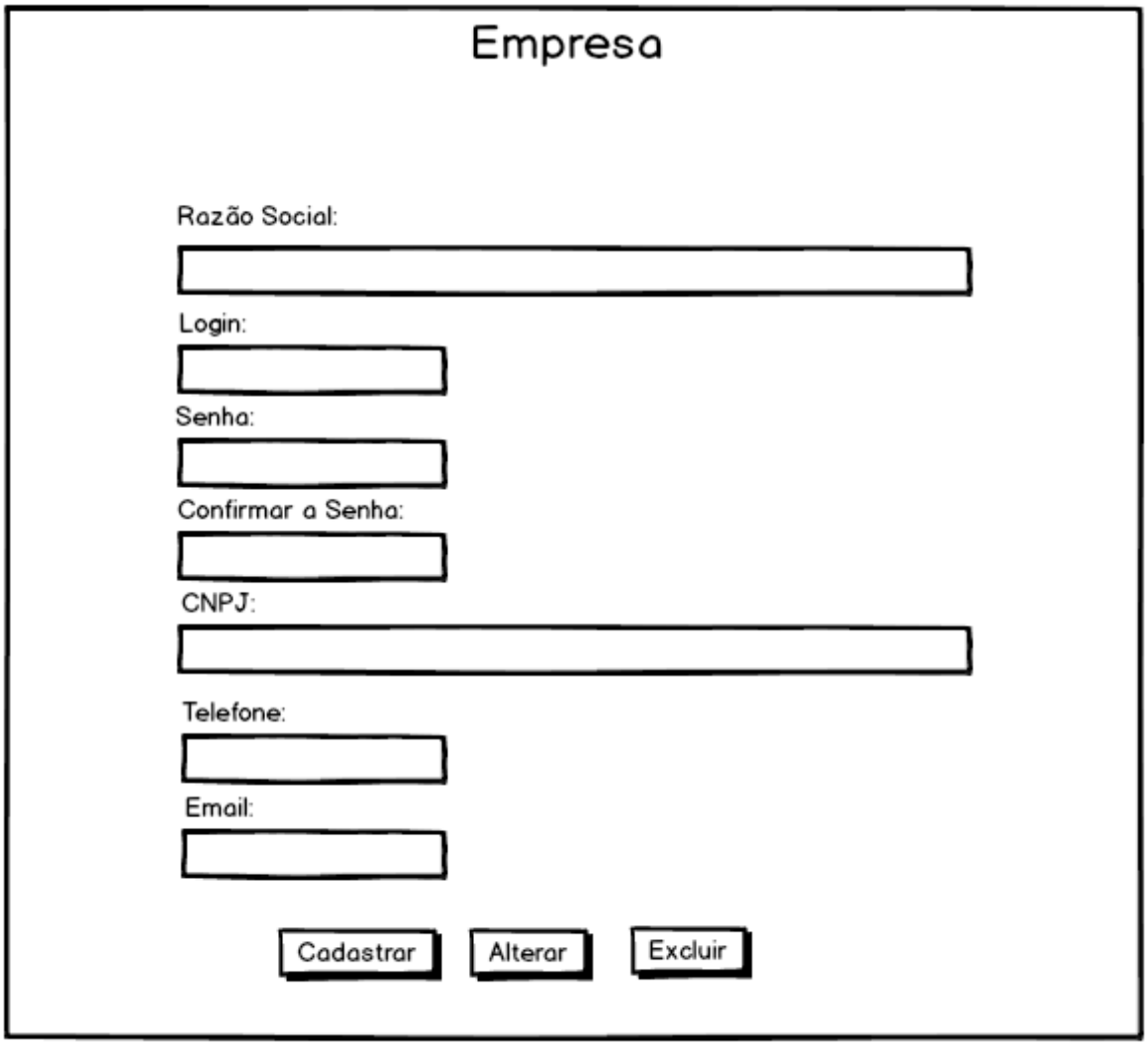

Figura 7: Protótipo de Tela de cadastro de Empresa

Manter Cidade

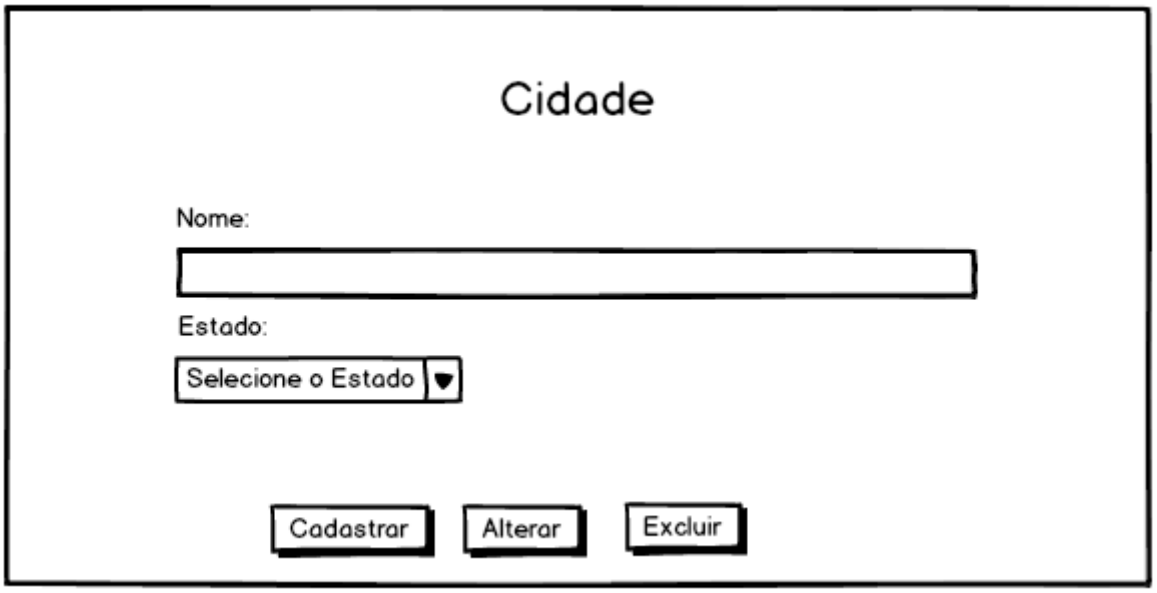

Figura 8: Protótipo de Tela cadastro de Cidades

Manter Estado

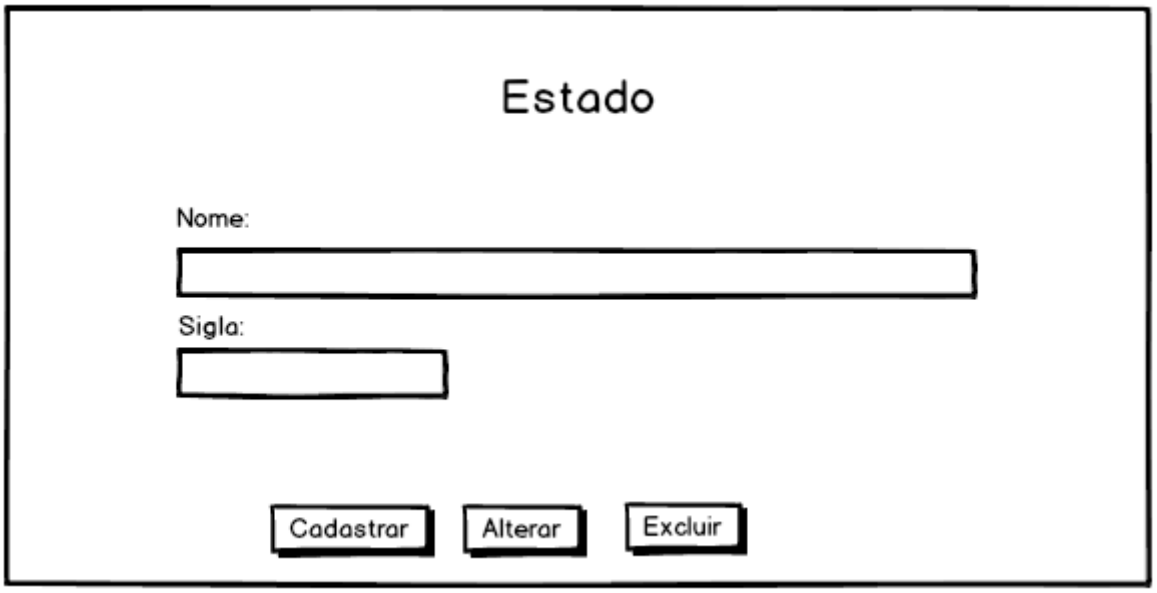

Figura 9: Protótipo de Tela cadastro de Cidades

Manter Opções

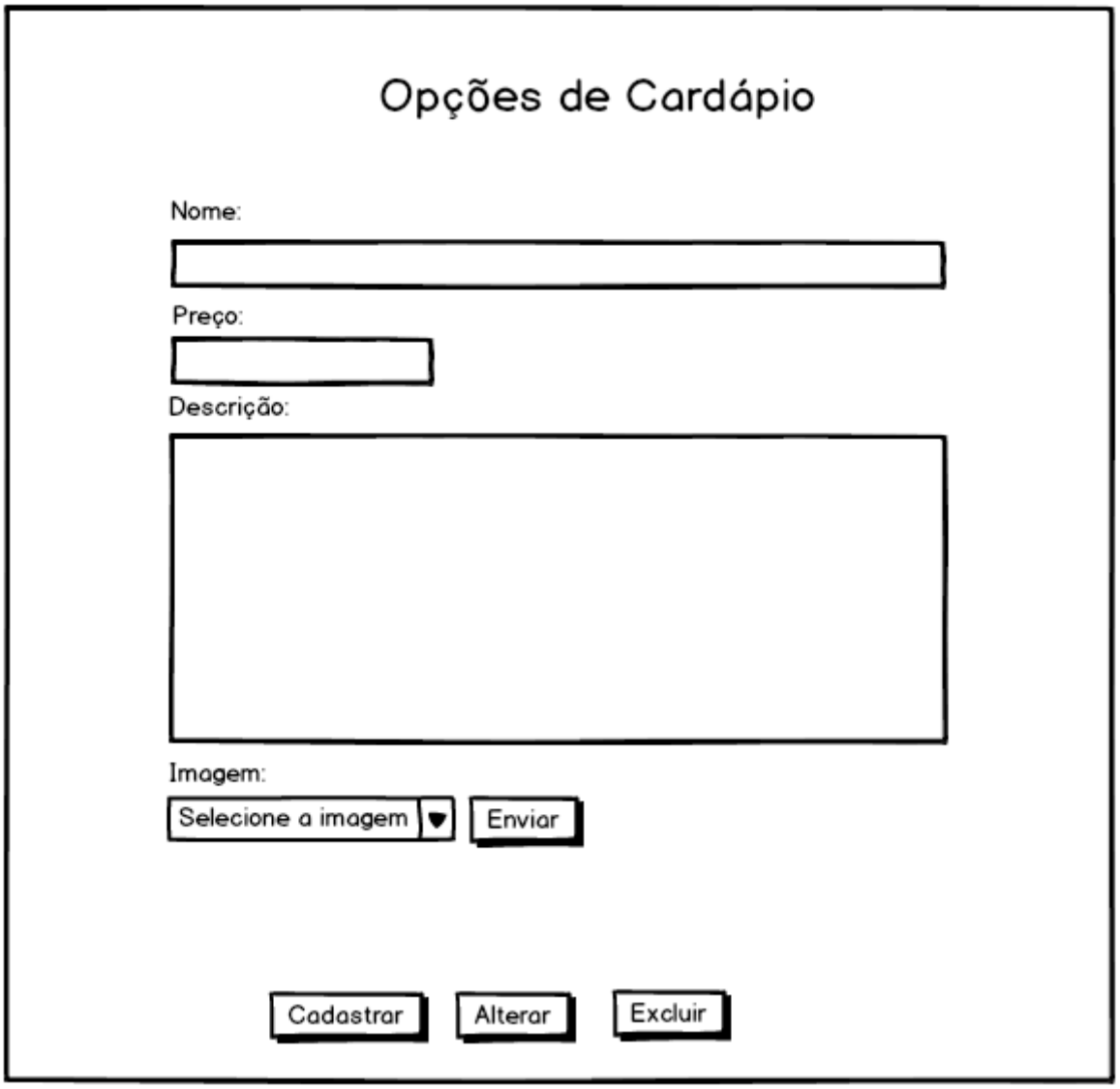

Figura 10: Protótipo de Tela cadastro de Cidades

Catálogo de Opções

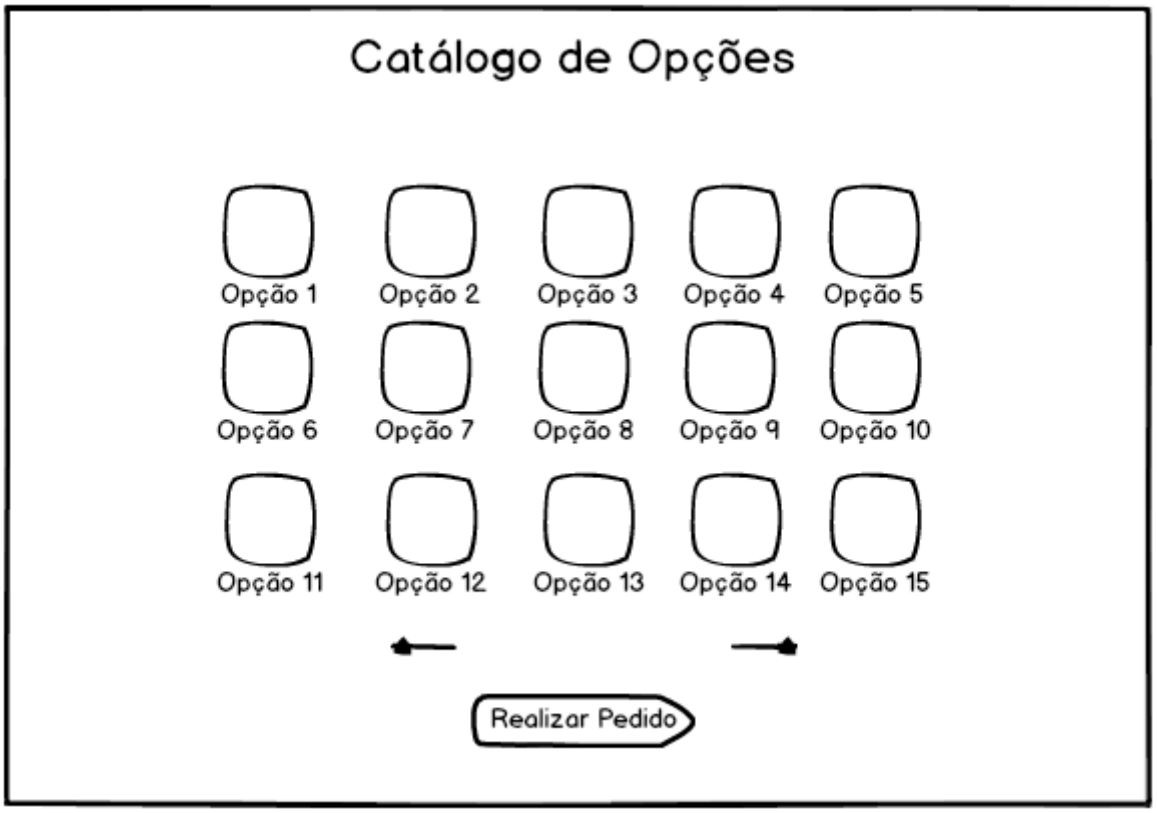

Figura 11: Protótipo de Tela de Opções

Pedidos Realizados

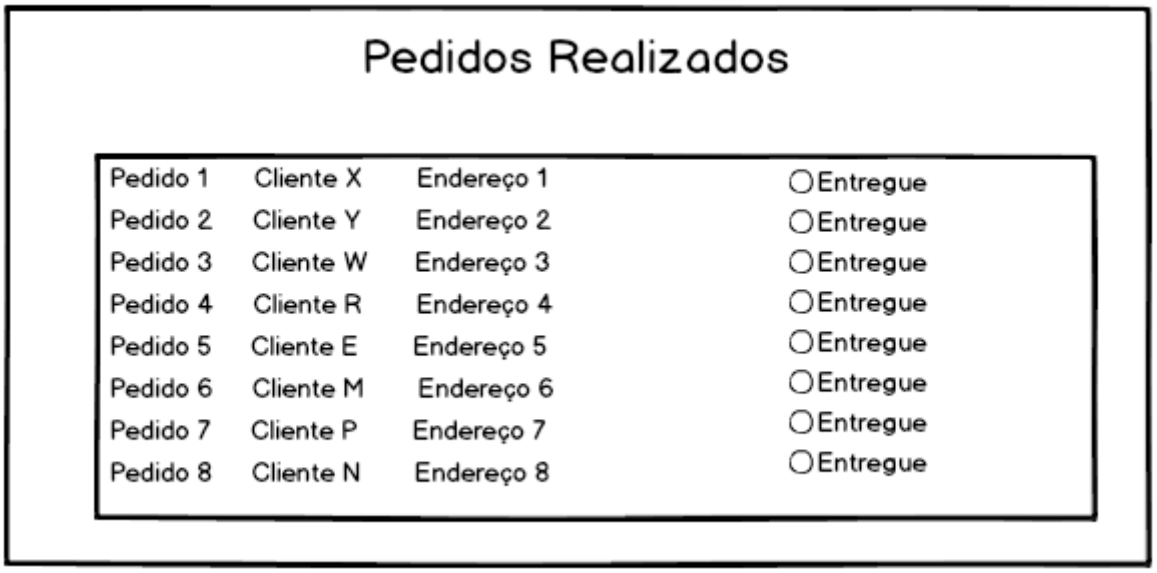

Figura 12: Protótipo de Tela de Pedidos Realizados

Realizar Pedidos

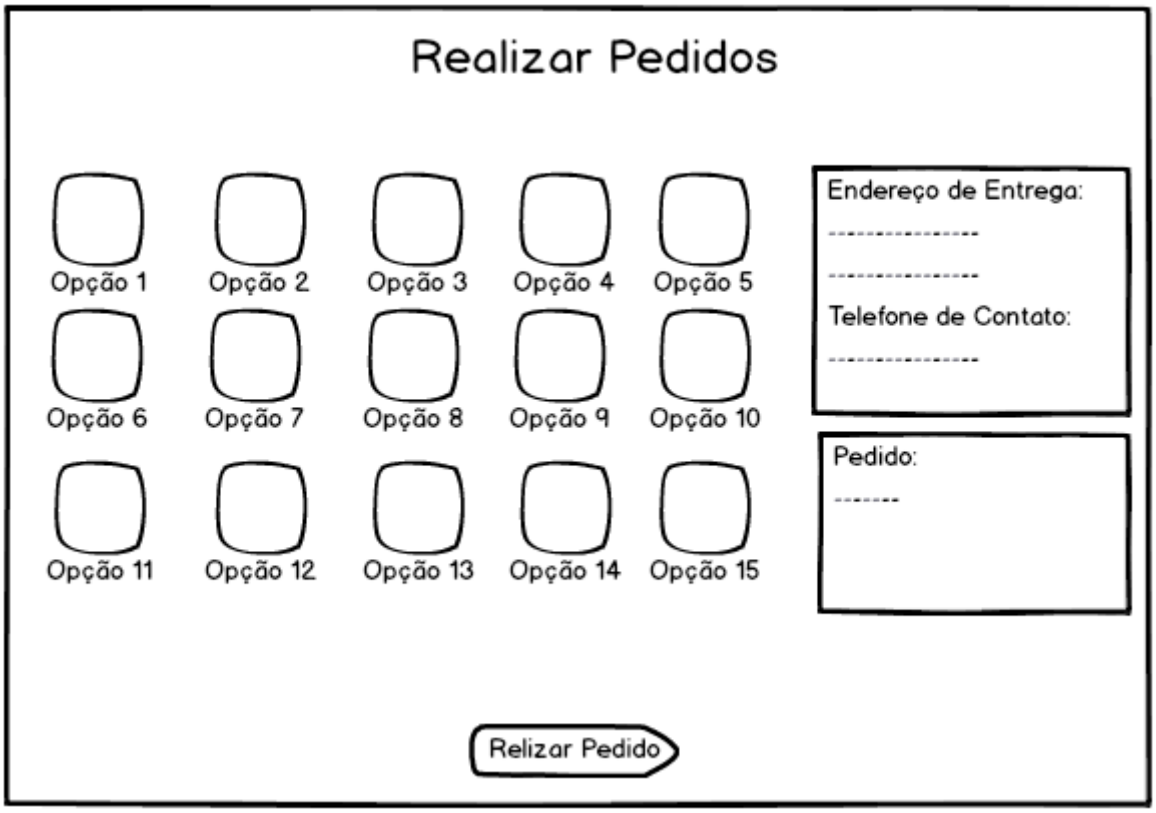

Figura 13: Protótipo de Tela de Realizar Pedidos

4 AVALIAÇÃO DE USABILIDADE

A usabilidade consiste em utilizar-se um produto e identificar se é fácil e se é de fácil aprendizagem, se grava em nossa memória, dificilmente ocorrem erros por parte do produto, se traz grande satisfação para quem usa e se atende com qualidade o que foi proposto. Então nos baseamos no quão rápido o usuário aprende com o menor ajuda possível, sendo assim intuitivo, e realiza suas tarefas sem dificuldade ou erros para avaliar a qualidade do produto. Segundo a norma ISO/IEC 9126, o termo usabilidade tem a seguinte definição:

"Capacidade do produto de software de ser compreendido, aprendido, operado e atraente ao usuário, quando usado sob condições especificadas."

São enquadradas quatro características que um produto de software deve possuir:

Inteligibilidade: A facilidade do usuário para compreender o funcionamento do software;

Apreensibilidade: A facilidade do usuário para aprender a utilizar o software;

Operacionalidade: A facilidade do usuário para operar o software;

Atratividade: O quão o software é atraente ao usuário.

Usabilidade pode ser relacionada à facilidade de uso de um software, e é uma dos fatores que são usados como instrumentos para uma avaliação de qualidade. Na usabilidade é avaliada qual é a satisfação do usuário em relação à interação com o produto.

Uma interface que agrade mais e acabe transmitindo um sentimento menor de frustação em relação ao seu uso é uma interface que possui consistência (Nielsen, 2000, 2002; Pearrow, 2000). Quanto mais simples e com um comportamento e visual singular para todo o sistema torna ele mais fácil de ser usado e deixa ele dentro das definições de consistência. No sistema operacional Windows 8 possuímos uma padronização relacionada a outros Windows anteriores, por isso vemos uma semelhança muito grande na apresentação visual da área de trabalho, do Windows Explorer e alguns painéis de configurações, até para não causar total estranhamento

entre usuários antigos, ao se tratar de interface Metro, vemos que a Microsoft apresentou um novo padrão visual que também vai aos aplicativos voltado para ela, vemos na figura a seguir a tela de entrada após acessar o sistema.

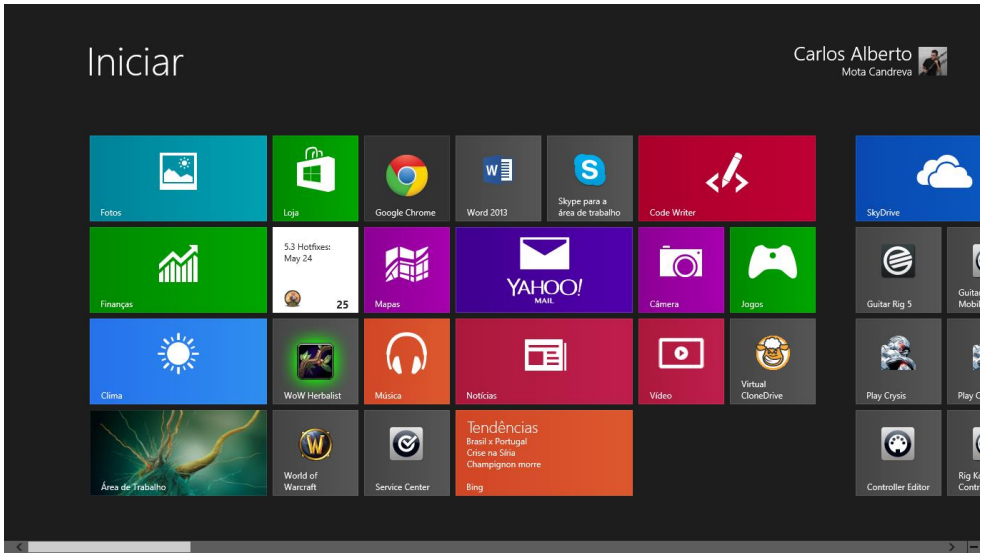

Figura 14: Tela Inicial da Interface Metro

A apresentação visual é toda intuitiva, para quem nunca teve contato com smartphones recentes pode parecer um pouco confuso, um dos riscos que a interface acaba correndo com sua proposta de atingir também o mercado de dispositivos móveis como tablets, pois como é toda voltada para telas sensíveis ao toque usuários que não possuem tais dispositivos podem ficar um pouco perdidos sem a sua habitual área de trabalho. Porém, alguns minutos depois o usuário percebe e entende a questão dos blocos dinâmicos (como são chamados os blocos de tamanhos diferentes apresentados na interface, ou no inglês Lives Tiles): ao tocar/clicar executa a ação que está descrita no próprio bloco. Por exemplo se clicarmos no bloco escrito Área de Trabalho, iremos direto até a Área de Trabalho.

Pode parecer sem importância, mas escolher a cor adequada é o principal meio de comunicação direta com o usuário, a cor pode interferir nos sentidos, emoções e padrão de pensamento de uma pessoa (Marcus, 1987). Utilizar as cores certas pode agilizar as ações do usuário e faz com que ele entenda claramente a informação

correta. A cor usada de maneira incorreta torna a informação incompleta, confusa ou incompreensível.

Apesar do usuário manter controle sobre a cor do fundo da tela inicial da interface Metro, ele não pode alterar fora das cores sugeridas pelo sistema, criando um padrão de cores para não confundir o usuário. Os aplicativos voltados completamente para a interface, ou seja, que ao executarem continuam na interface e não vão para o modo Área de Trabalho, mantém uma sintonia de cores, tendo cada um sua cor estabelecida trazendo uma noção de onde o usuário está a partir da cor associada ao aplicativo que ele clicou. São usadas cores vivas e que remetem a função do aplicativo e trazem uma identidade visual ao sistema, como temos a tendência de associar cores a situações de nossas vidas, é interessante utilizar cores para indicar diversas situações. Claro que cada desenvolvedor tem sua liberdade na escolha da cor, mas dificilmente veremos um desenvolvedor que irá escolher uma cor sem antes analisar a situação que aquela cor remete ou se a cor está associado ao seu produto, posso utilizar de exemplo o aplicativo de e-mails do Yahoo que utiliza a cor roxa que é uma característica da empresa desde seu início criando assim a identidade visual da marca, segue a imagem.

Figura 15: Exemplo do Aplicativo Yahoo com sua cor característica

A falta de um menu de ajuda de fácil acesso na interface Metro prejudica um pouco seu fácil aprendizado e alguns comandos não são totalmente intuitivos, como por exemplo chamar uma barra de ação ao clicar com o botão direito em suas

aplicações, por ela ser desenvolvida pensando também em dispositivo sensíveis ao toque ela acabou criando está pequena confusão para quem utiliza com teclado e mouse. Porém dois aspectos agradáveis facilmente notados são: nenhum botão é utilizado para mais de uma função diferente, ou seja, não teremos comportamentos inesperados nesse sentido, se clica em um botão que foi identificado como pesquisa ele executa a função de pesquisa; o outro aspecto agradável é que os blocos mostram informações dinâmicas sobre o conteúdo deles, como o aplicativo de notícias que fica apresentando as notícias mais recentes vindas da internet, além de possuir um nome totalmente intuitivo sobre sua função. Claro que ai voltamos também a questão que cada desenvolvedor tem a liberdade de trabalhar seu aplicativo como achar mais adequado.

Imagens e ícones são excelentes para as pessoas que nunca utilizaram, pois ajudam a entender melhor as ações que serão executadas, a interface Metro tenta aproveitar o máximo disso, trazendo em poucas imagens ou ícones qual será a ação efetuada pelo o que se está mexendo, trazendo pouco ou nenhum texto junto, mas sendo totalmente intuitivo.

Interfaces planejadas da melhor maneira permitem acelerar o processo de interagir com o conteúdo, permitindo usar teclas de atalho e de comandos introduzidos pelo teclado, que permitem uma flexibilidade e não só um modo de interação e tornando para usuário mais experientes que saibam usar o processo mais rápido (Foley, 1990). A interface Metro possui uma completa integração de busca que a qualquer momento que digitamos dentro da interface acessa automaticamente o perfil de busca que permite busca dentro dos arquivos no computador, dentro de funções do sistema e dentro dos próprios aplicativos, trazendo uma imensa facilidade para quem está usando o sistema, segue uma figura exemplando.

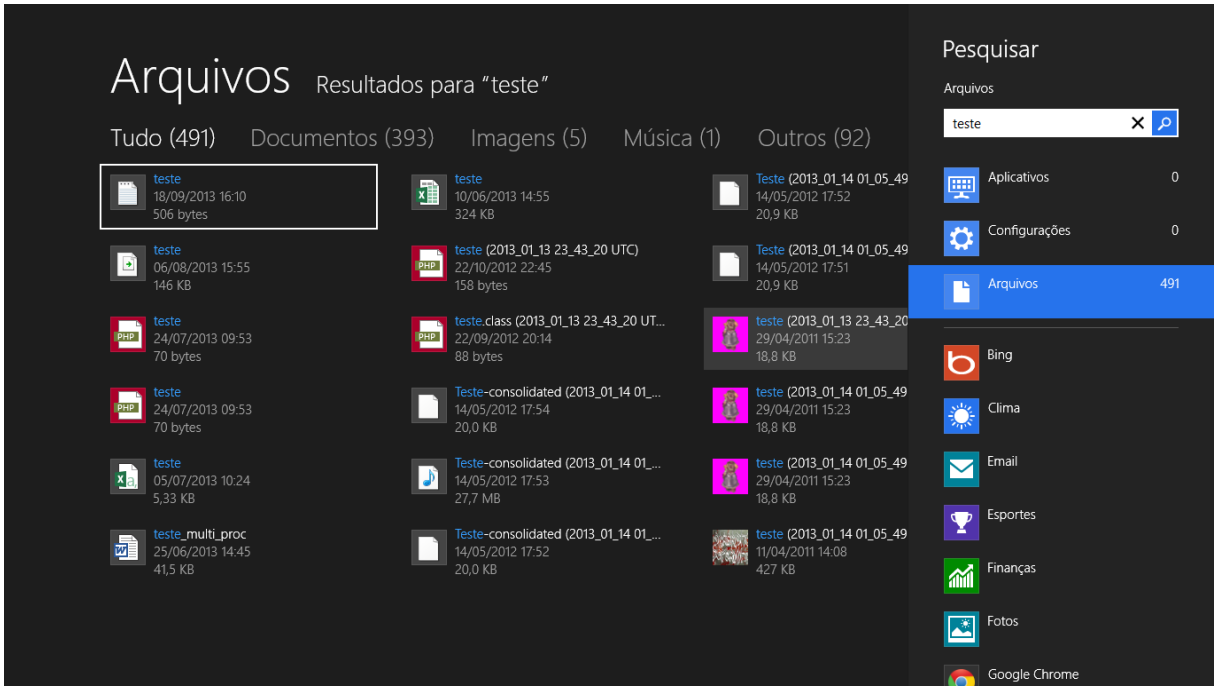

Figura 16: Sistema de Buscas integrado a interface Metro

Todo conhecimento que o usuário já adquiriu a respeito de tudo que faz parte do seu cotidiano, usar metáforas possuam coisas já conhecidas pelo usuário deixando a interação com o sistema mais intuitivo (Apple, 1992).

Quem acessa a interface Metro tem pouco esforço para memorizar os aspectos da interface, pois é totalmente intuitiva e autoexplicativa, são poucos comandos utilizados, sendo em sua grande maioria de fácil memorização, tornando a tarefa tão simples que parece natural (Foley, 1990). A que foi planejada e foi realizada a escolha correta dos ícones, imagens e elementos que irão auxiliar o uso do usuário à interface, é uma interface de boa qualidade (Pressman, 1992).

Uma maneira eficiente de uma pessoa acessar as funções que não são constantemente usadas é o uso de barras de menus que irão possuir essas ferramentas. Seguindo este modelo ao apertarmos o botão direito do mouse aparece uma barra com ferramentas que podemos usar naquele momento dentro da aplicação ou tela.

Quando conseguimos controlar os objetos representados no computador e o objeto manipulado permanecer visível enquanto sobre ele estiver realizada uma operação e o efeito dessa manipulação deve ser imediatamente notado, chamamos de manipulação direta (Apple, 1992). Quando alteramos na tela inicial um bloco dinâmico ele fica sobre efeito da manipulação direta e funciona sem qualquer surpresa ou comportamento inesperado.

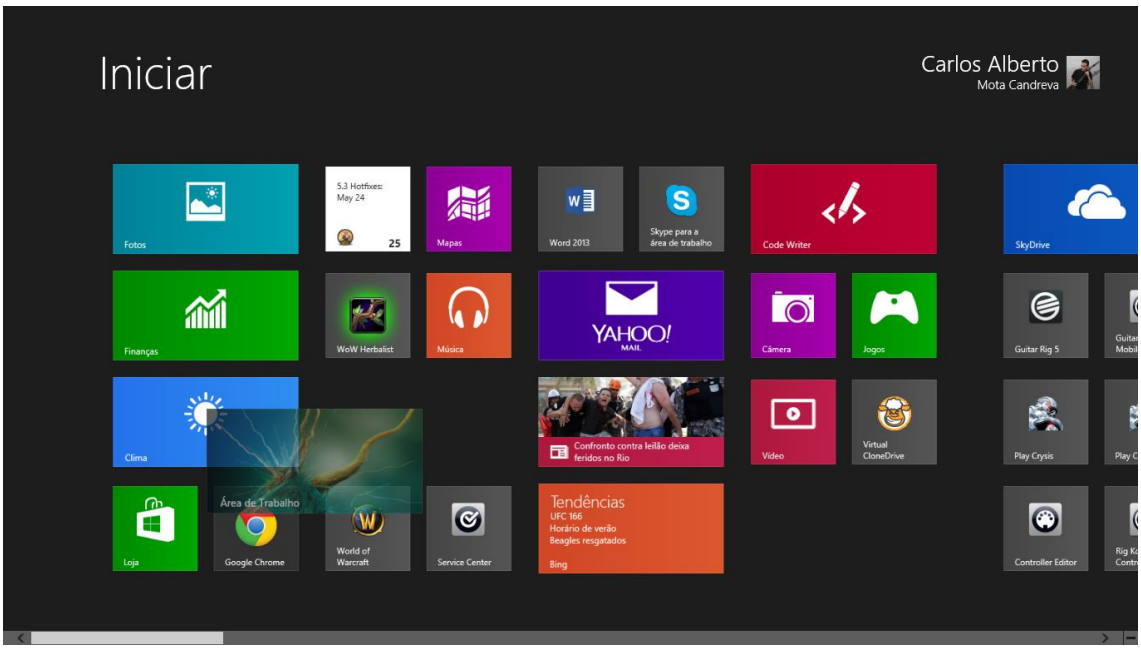

Figura 17: Efeito de manipulação direta em um bloco dinâmico

Para aumentar a facilidade de usa, a interface deve possuir apenas informações importantes ao contexto que está mostrando atualmente (Pressman, 1992). Apesar de termos informações bem enxutas e estar bem adequada a essa regra, infelizmente, acaba errando em ser pouca informação mostrada, tornando um pouco difícil a vida de quem está trabalhando pela primeira vez com a nova interface do Windows.

A customização é bem grande na interface Metro, o usuário pode escolher a ordem dos blocos dinâmicos, pode alterar o fundo, as cores gerais, pode escolher o tamanho dos blocos e ajustas conforme sua necessidade na tela inicial. Dentro de cada aplicativo fica a cargo do desenvolvedor, porém seguir o padrão proposto pela Microsoft é tornar cada vez mais seu software consistente e intuitivo, pois o usuário já estará adequado ao comandos e terá uma facilidade muito mais maior com seu sistema

4.1 ESTUDO DE CASO

O desenvolvimento de uma aplicação para Windows 8 é cercado de pequenos detalhes. A Microsoft não permite alteração direta de informações do computador via aplicativo, mantendo o aplicativo sendo executado dentro de uma área de segurança, tornando assim o sistema operacional mais seguro e com menos chances de falha por parte de aplicativos de terceiros. Para utilização de fonte de dados é necessário o uso de WebServices.

Toda permissão sobre o que a aplicação irá poderá executar, como conectar-se a internet, ou receber algum arquivo vindo do computador deve ser declarado Package.appxmanifest, ele possui todas as informações de como o aplicativo irá interagir diretamente com o sistema operacional.

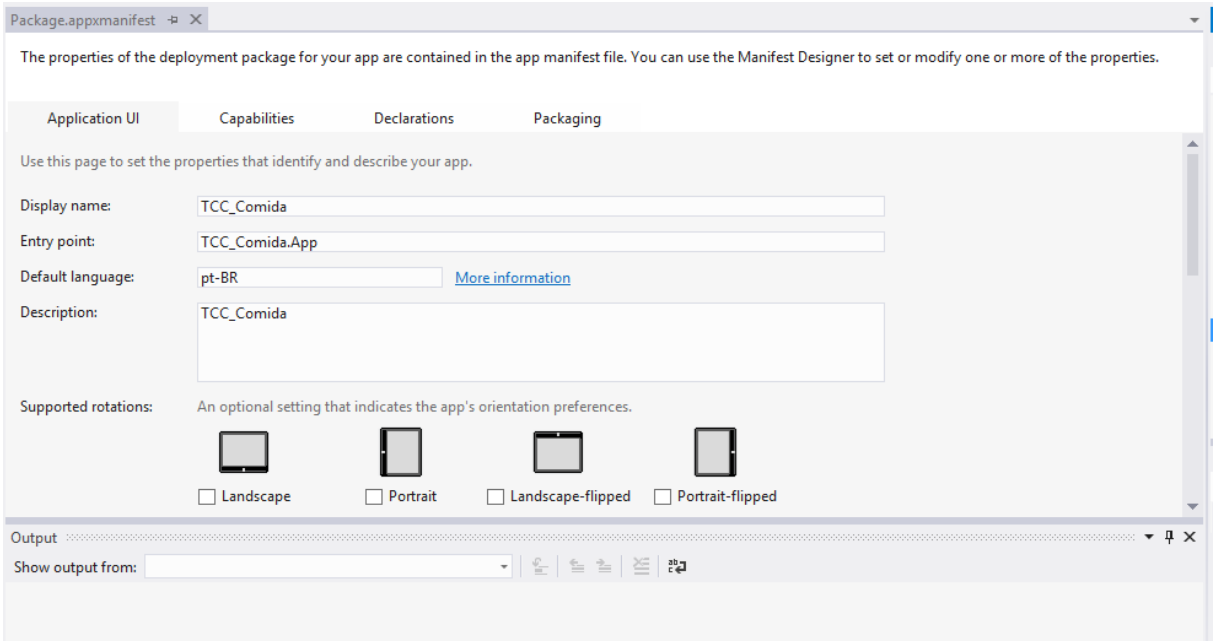

Figura 18: Package.appxmanifest

Vemos na figura por exemplo interações de interface como: Qual será o nome exibido, sua descrição, a linguagem padrão e por exemplo qual tipo de exibição de orientação ele irá executar, caso você precise que seu aplicativo seja fixado a horizontal por exemplo.

Na aba Capabilities, declaramos se ele poderá utilizar algumas pastas especificas do computador de maneira limitada, se ele poderá ter iteração com a

internet, com dispositivos de armazenamento removíveis entre outras opções, esta aba define de maneira geral as capacidades de seu aplicativo.

A aba Declarations possuímos opções de informar quais arquivos ele poderá receber, se receberá certos tipos de conteúdo ou não, por exemplos tipos de imagem, essa aba define o que seu aplicativo poderá receber ou não, ou seja declaramos que ele é capaz de acessar a pasta de imagens em Capabilities e aqui declaramos que ele será permitido salvar imagens na pasta de imagens do usuário.

E então na última aba, Packaging, fazemos as declarações em relação ao logo do aplicativo, seu nome e sua versão, é aba que faz as definições de empacotamento em aplicativo do que foi programado.

A programação do aplicativos recebe alguns tratamentos diferentes de se programar para Web utilizando C#, ou programar desktop utilizando C#, sua aplicação poderá ser tanto Web quanto desktop, então não vemos tantas diferenças na hora de programar, mas devemos nos atentar a claro as declarações feitas no manifesto.

O visual é todo programado em XAML, sigla de eXtensible Application Markup Language, é uma linguagem declarativa baseada no XML, ela não pode executar códigos por si própria, por isso o C# cuida de todo tratamento de programação por traz do XAML. No caso do aplicativo desenvolvido vou utilizar um exemplo de código onde busco as informações via WebService, utilizando XAML, C# e PHP.

O componente do tipo ListView é declarado no XAML:

```
<ListView x:Name="lstEstados" Margin="152,448,121,0" Grid.Row="1" VerticalAlignment="Top"
Height="150" HorizontalContentAlignment="Right" >
            <ListView.Projection>
                <PlaneProjection/>
            </ListView.Projection>
            <ListView.ItemTemplate>
                <DataTemplate>
                    <Grid Width="500" VerticalAlignment="Center"
HorizontalAlignment="Center">

                        <Grid.ColumnDefinitions>
                            <ColumnDefinition Width="120" />
                            <ColumnDefinition Width="*" />
                            <ColumnDefinition Width="150" />
                        </Grid.ColumnDefinitions>


                        <TextBlock Grid.Column="0" Text="{Binding codEstado}" />
                        <TextBlock Grid.Column="1" Text="{Binding est_nome}" />
                        <TextBlock Grid.Column="2" Text="{Binding est_sigla}" />
                    </Grid>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
```

Código 1: Exemplo de objeto ListView em linguagem XAML

As colunas serão alimentadas via C#, a indicação de onde vêm o retorno está no começo do XAML:

No C# então nos comunicamos com o WebService desenvolvido para trazer as informações do banco:

```
JsonWebAsync.JsonWebClient client = new JsonWebAsync.JsonWebClient();
            HttpWebRequest req =
HttpWebRequest.CreateHttp("http://localhost/webServicesTCC/SlimTCC/estados");
            req.ContentType = "application/json";


            var resp = await client.DoRequestJsonAsync<EstadoResponse>(req);
            lstEstados.ItemsSource = resp.estados;
```

Código 2: Chamada de WebService via C#

Aqui para simplificar a requisição utilizo uma classe externa criada para manipular requisições no formato JSON, JavaScript Object Notation, é uma formatação leve de troca de dados. Foi criado um projeto que executa em conjunto com o sistema chamado JsonWebAsync, que possui todos os métodos para requisição de WebServices rest com resposta no padrão Json, no caso o método usado foi DoRequestJsonASync<>:

```
    public async Task<T> DoRequestJsonAsync<T>(WebRequest req)
       {
           var ret = await DoRequestAsync(req);
           var response = await ret.ReadToEndAsync();
           return Newtonsoft.Json.JsonConvert.DeserializeObject<T>(response);
       }
```

Código 3: Método assíncrono para busca de Json via WebService

Nele basicamente é feita uma requisição via web de um link parâmetro, no meu caso local, pois criei um servidor apache local para facilitar a programação, mas poderia ser o link do servidor apache que será utilizado para sua aplicação, a resposta é recebida em JSON e convertida numa resposta em objeto para mais fácil compreensão do ListView, que por padrão não lê JSON.

Para o desenvolvimento do WebService em PHP utilizei do framework preparado para este tipo de requisições o Slim Framework. Ele torna a programação de WebServices intuitiva e prática. Vou exemplificar de maneira simples como funciona o código em si. No início do código é declarado o tipo de retorno que o aplicativo realizará e o caminho base:

```
require '../Slim/Slim.php';

\Slim\Slim::registerAutoloader();

$app = new \Slim\Slim();

//$app->response()->header('Content-Type', 'application/xml');

$app->response()->header('Content-Type', 'application/json;charset=utf-8');

$app->get('/', function () {

echo "SlimTcc";

});
```

Código 4: Inicialização do Web Service em PHP utilizando o SlimFramework

Para definir o retorno em JSON, utilizamos do 'application/json;charset=utf-8', falando o tipo do arquivo retornado e o tipo de caracteres aceitos. Logo após definiremos as rotas que irão dar acesso aos nosso métodos, usarei como exemplo os métodos relacionados a tabela de estados.

Declaramos aqui por exemplo, que quando a requisição ser de um método post, utilizado o link '/estados' será utilizado o método AddEstado, que incluí um estado, mas caso possuir um id no fim do link, '/estados/:id' vindo de uma requisição post ele irá executar o método AlteraEstado.

Como exemplo vou mostrar o funcionamento do método de inclusão de um estado:

```
function addEstado()
{
$request = \Slim\Slim::getInstance()->request();
$estado = json_decode($request->getBody());
$sql = "INSERT INTO estado (est_nome,est_sigla) values (:est_nome,:est_sigla) ";
$conn = getConn();
$stmt = $conn->prepare($sql);
$stmt->bindParam("est_nome",$estado->est_nome);//nome vindo do post do formulário do app
$stmt->bindParam("est_sigla",$estado->est_sigla);
$stmt->execute();
$estado->codEstado = $conn->lastInsertId();
echo json_encode($estado);
}
```

Código 5: Exemplo de método de inclusão no WebService

O método recebe o JSON vindo do aplicativo, e executa uma SQL baseada nas informações vindas, executa e retorna um JSON com o estado incluído e seu código gerado capturado pelo método '$conn->lastInsertId();'. Também é utilizado para conexão com o banco de dados o método 'getConn();'.

```
function getConn()
{
return new PDO('mysql:host=127.0.0.1;dbname=apptcc',
'root',
'',
array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8")
);
```

Código 6: Exemplo de Conexão no WebService

Ele possui a definição do tipo do banco de dados e a conexão realizada para obtermos os dados.

No geral, aplicativos desenvolvidos para Windows Store são feitos com a integração do WebService via Windows Azure, a plataforma na nuvem da Microsoft, porém como é um ferramenta totalmente paga inviabilização a utilização para o estudo de caso. Neste problema encontrado a solução foi criar um WebService via uma alternativa livre, utilizando do PHP, um servidor Apache e um banco de dados MySQL. O Slim Framework é um framework aberto que permite facilmente criar webservices de diversos padrões, o utilizado no estudo foi o rest, com retorno em JSON. Está se mostrou a solução mais prática para substituição do Windows Azure.

4.2 TELAS EXEMPLO DO APLICATIVO FINALIZADO

Agora alguns exemplos das telas na versão final do aplicativo serão apresentados. Nem todas as telas estarão presentes.

Figura 19: Tela de Login

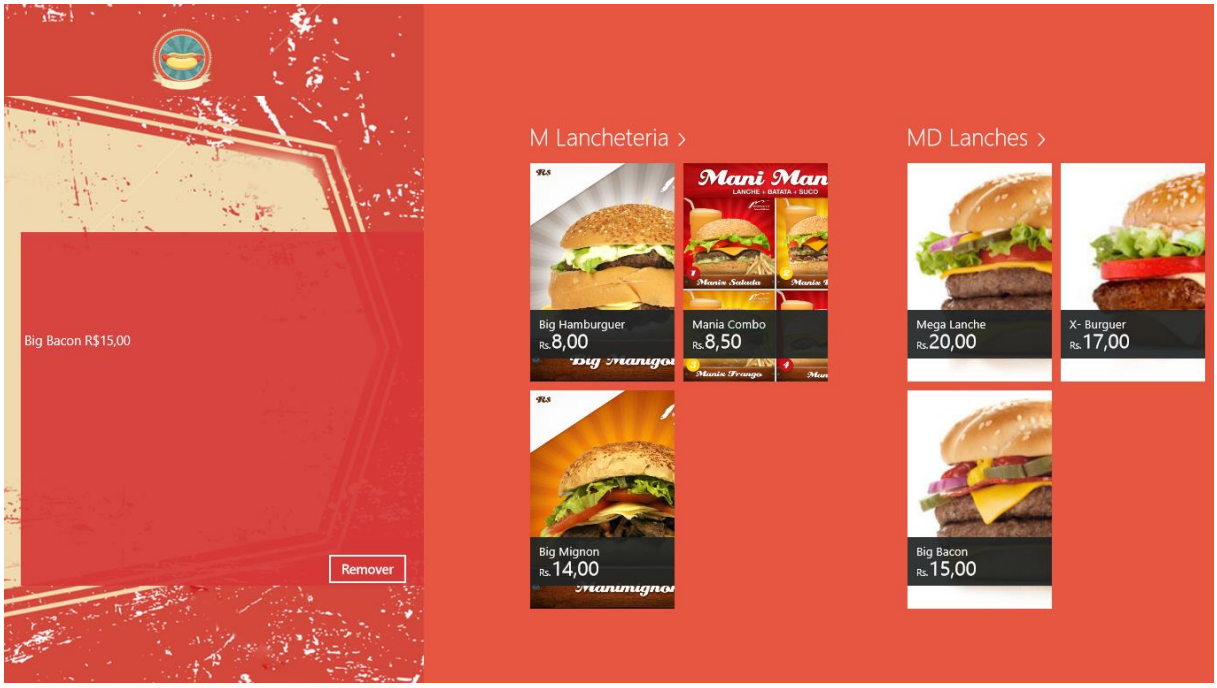

Figura 20: Tela Principal

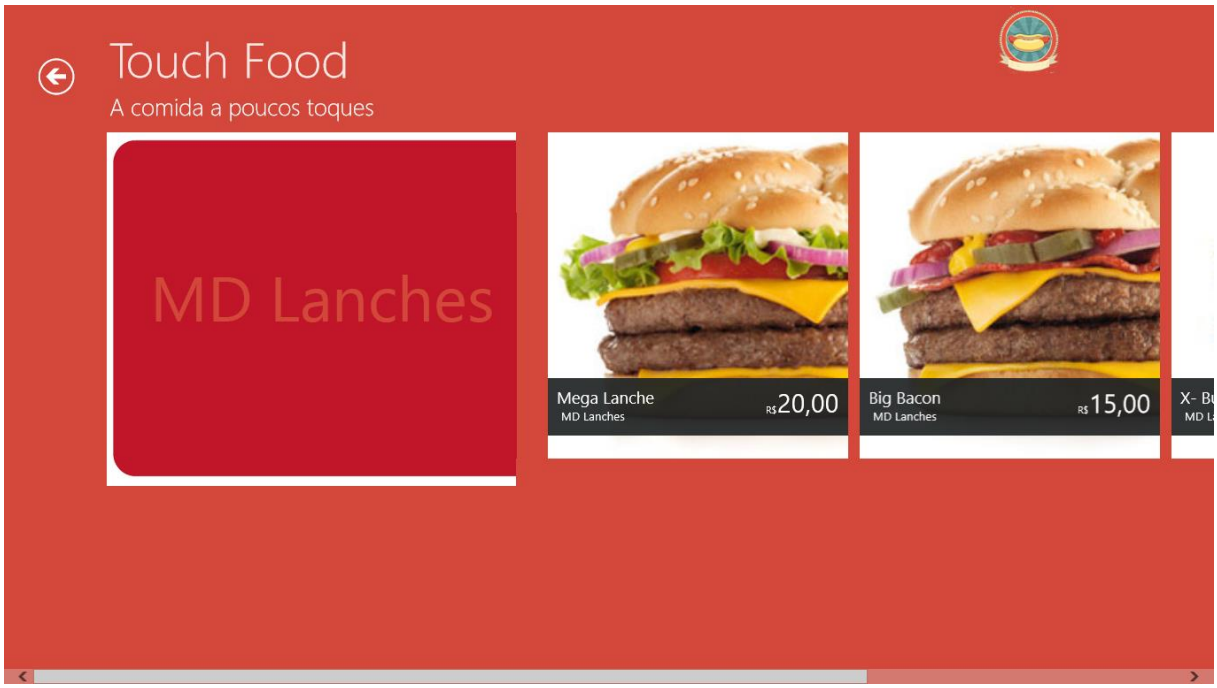

Figura 21: Catalogo de opções da empresa.

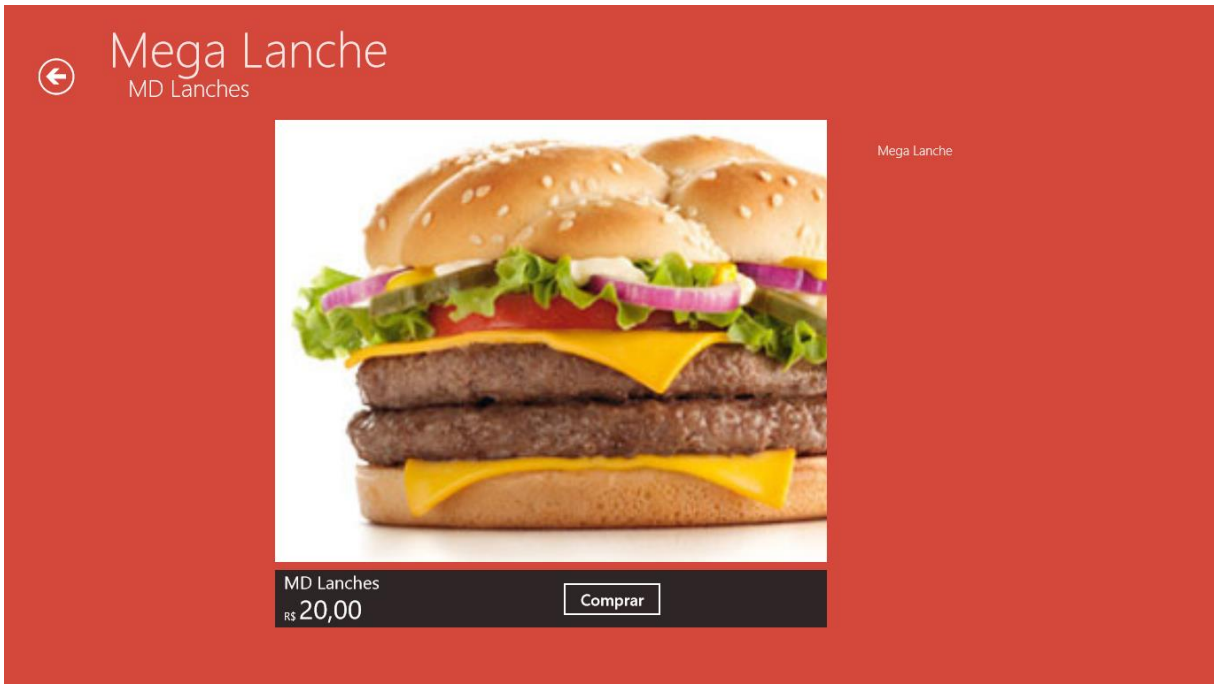

Figura 22: Opção selecionada

5 CONCLUSÕES

O Windows 8 traz em si uma pesada missão: trabalhar em duas vertentes de consumidores comuns, os que aderiram dispositivos móveis e os que se mantem nos PCs e Notebooks comuns. Essa proposta dupla acaba acarretando dificuldades na maneira de padronizar uma versão que funcione nas duas, principalmente em termos de layout e usabilidade, já que decidiu-se fazer algo que fosse muito compatível com as propostas já feitas em dispositivos móveis.

Porém essa arriscada missão é necessária, pois a Microsoft já perdeu uma grande fatia do mercado móvel e está tentando entrar tardiamente no mercado, e uma das tendências é a substituição dos PCs e Notebooks serem cada vez mais substituídos por tablets por consumidores mais casuais que não exigem tanto processamento de seus equipamentos.

O Windows 8 atendeu muito bem a definição da norma ISO/IEC 9126, onde ele é facilmente compreendido, operado e é atraente ao usuário. Claro, o uso com os tradicionais mouse e teclados torna menos intuitivo, pois a interface Metro é claramente planejada para dispositivos sensíveis ao toque que estão agora em grande quantidade no mercado, desde monitores até tablets. Porém, em termos de Inteligibilidade, Apreensibilidade, Operacionalidade e Atratividade, ele atendeu com grande êxito como demonstrado.

O estudo de caso produzido acabou gerando algumas dificuldades em relação a conexão com a base de dados a um servidor externo, porém atende de maneira intuitiva as premissas propostas na usabilidade planejada pela Microsoft, apesar de ter sido um risco que a Microsoft corre trazendo algo tão novo ao mercado tão consolidado de sistemas operacionais para computadores de mesa, ela acaba mostrando as portas das tendências antes futuristas que agora se tornam mais e mais realidade, e apesar de concorrido mercado de sistemas operacionais portáteis a Microsoft pode sim conquistar grande fatia já que ao mesmo tempo que agrada quem já utiliza o sistema da empresa em seu desktop ou notebook e agrada também quem irá utilizar em dispositivos móveis com apenas um sistema operacional, trazendo então uma grande satisfação em seus consumidores.

REFERÊNCIAS

APPLE COMPUTER. Macintosh human interface guidelines. New York: Addison-Wesley, 1992.

DAQUINO FERNANDO. Metro UI: a interface que dominará o Windows. TecMundo, 13 de Junho de 2011. Disponível em: < http://www.tecmundo.com.br/microsoft/10736-metro-ui-a-interface-que-dominara-o-windows.htm>. Acesso em: 10 de Janeiro de 2013.

DE SOUZA ELSON. Review do Windows 8.1 Preview: sistema evolui, mas Microsoft deve inovar mais. TechTudo, 27 de Junho de 2013. Disponível em:< http://www.techtudo.com.br/artigos/noticia/2013/06/review-do-windows-81-preview-sistema-evolui-mas-microsoft-deve-inovar-mais.html>. Acesso em: 28 de Junho de 2013.

FOLEY, J. D. et al. Computer graphics, principles and practice. New York: Addison-Wesley, 1990.

KEENE JAMIE. Windows Store launches for Windows 8. The Verge, 29 de Fevereiro de 2012. Disponível em:<http://www.theverge.com/2012/2/29/2832684/windows-8-store-announcement-mwc>. Acesso em: 14 de Março de 2013.

MARCUS, A. Color: a tool for computer graphics communication. In: Color in computer graphics. California, SIGGRAPH, 1987. Tutorial notes, n. 24, p. 3-9.

MIKKO TIKKANEN. Dissection of Windows 8 Metro UI – 8 features. MintUsability, 6 de Março de 2012. Disponível em: <http://mintusability.com/blog/dissection-of-windows-8-metro-ui-8-features/>. Acesso em: 10 de Janeiro de 2013.

NATE RALPH. Windows 8 Metro UI: A Bold New Face for Windows. PCWorld, 5 de Março de 2012. Disponível em: < http://www.pcworld.com/article/251340/windows_8_metro_ui_a_bold_new_face_for_windows.html >. Acesso em: 30 de Dezembro de 2012.

NEVES PEDRO M. C. e RUAS RUI P. F. O Guia Prático Do MySQL. Lisboa, 2005.

NIELSEN, J. Designing web usability. Indianapolis: News Riders Publishing, 2000

NIELSEN, J.; TAHIR, M. Homepage: usabilidade - 50 websites desconstruídos. Rio de Janeiro: Campus, 2002.

OLSON PHILIP. Manual do PHP. PHP,1997-2013. Disponível em:< http://www.php.net/manual/pt_BR/index.php>. Acesso em: 22 de Outubro de 2013.

PEARROW, M. Web site usability handbook. Massachusetts: Charles River Media, 2000.

PRESSMAN, R. S. Software engineering: a practioner's approach. 3. ed. New York: McGraw-Hill, 1992.

RIZZO THOMAS. Pro SQL Server 2005. Nova Iorque: Apress, 2005.

SANT'ANNA MAURO. Windows 8 do ponto de vista do desenvolvedor. Linha de Código, Maio de 2011. Disponível em:< http://www.linhadecodigo.com.br/artigo/3288/windows-8-do-ponto-de-vista-do-desenvolvedor.aspx>. Acesso em: 20 de Maio de 2013.

TROELSEN ANDREW. Pro C# and the .NET Plataform. 5ª Edição, Nova Iorque: Apress, 2010.