

Diferentes perspectivas no estudo de sistemas de informação: estudos de caso



Organizadores:

Sheyla Natália de Medeiros

Fábio Nicácio de Medeiros

ISBN: 978-85-5597-039-9

Diferentes perspectivas no estudo de sistemas de informação: estudos de caso

Sheyla Natália de Medeiros
Fábio Nicácio de Medeiros
(Organizadores)

Instituto de Educação Superior da Paraíba - IESP

Cabedelo
2017



INSTITUTO DE EDUCAÇÃO SUPERIOR DA PARAÍBA – IESP

Diretora Geral

Érika Marques de Almeida Lima Cavalcanti

Diretora Acadêmica

Iany Cavalcanti da Silva Barros

Diretor Administrativo/Financeiro

Richard Euler Dantas de Souza

Editora IESP

Editores

Cícero de Sousa Lacerda

Hercilio de Medeiros Sousa

Jeane Odete Freire Cavalcante

Josemary Marcionila Freire Rodrigues de Carvalho Rocha

Corpo editorial

Antônio de Sousa Sobrinho – Letras

Hercilio de Medeiros Sousa – Computação

José Carlos Ferreira da Luz – Direito

Marcelle Afonso Chaves Sodré – Administração

Maria da Penha de Lima Coutinho – Psicologia

Rafaela Barbosa Dantas – Fisioterapia

Rogério Márcio Luckwu dos Santos – Educação Física

Thiago Bizerra Fideles – Engenharia de Materiais

Thiago de Andrade Marinho – Mídias Digitais

Thyago Henriques de Oliveira Madruga Freire – Ciências Contábeis

Copyright © 2017 – Editora IESP

É proibida a reprodução total ou parcial, de qualquer forma ou por qualquer meio. A violação dos direitos autorais (Lei nº 9.610/1998) é crime estabelecido no artigo 184 do Código Penal.

O conteúdo desta publicação é de inteira responsabilidade do(os) autor(es).

**Dados Internacionais de Catalogação na Publicação (CIP)
Biblioteca Padre Joaquim Colaço Dourado (IESP)**

D569 Diferentes perspectivas no estudo de sistemas de informação: estudos de caso [recurso eletrônico] / organizadores, Sheyla Natália de Medeiros, Fábio Nicácio de Medeiros. - Cabedelo, PB : Editora IESP, 2017.

150 p.

Formato: E-book

Modo de Acesso: World Wide Web

ISBN: 978-85-5597-039-9

1. Sistemas de informação. 2. Business intelligence. 3. Tecnologia assistida. 4. Banco de dados. 5. Comércio eletrônico. I. Medeiros, Sheyla Natália. II. Medeiros, Fábio Nicácio de.

CDU 005.94:004.9

Bibliotecária: Angélica Maria Lopes Silva – CRB-15/23

Editora IESP

Rodovia BR 230, Km 14, s/n,
Bloco E - 3 andar - COOPERE
Morada Nova. Cabedelo - PB.
CEP 58109-303

Prefácio

"*Diferentes Perspectivas no Estudo de Sistemas de Informação - Estudos de Caso*" foi idealizado pelos professores e pesquisadores Sheyla Natália de Medeiros e Fábio Nicácio de Medeiros - membros do corpo docente dos cursos de Sistemas para Internet e Sistemas de Informação do Instituto de Ensino Superior da Paraíba –IESP.

Os conteúdos aqui presentes expressam a pluralidade de interesses que a área de Sistemas de Informação abarca, incluindo importantes discussões sobre, tecnologias assistivas, acessibilidade, protocolos, desenvolvimento, banco de dados e *business intelligence*. Traz contribuições de pesquisadores e estudantes, valorizando a produção local, mas, ao mesmo tempo, procurando um equilíbrio com artigos advindos de outras regiões do país e do mundo.

Iniciamos com uma pesquisa fundamentada nos conceitos de inclusão e acessibilidade na web que busca o desenvolvimento consciente de sites como e-commerce mostrando os impactos causados por interfaces despreparadas para o público daltônico e pontuando tecnologias assistivas existentes para auxiliar na melhoria desse desenvolvimento. Seguimos falando sobre o protocolo DDP (*Distributed Data Protocol*) de sistemas web que é um protocolo implantado na plataforma MeteorJS para que aplicações web estejam sempre atualizadas perante o usuário evitando inconsistência na informação. Há também uma pesquisa sobre o desenvolvimento de um sistema web utilizando o *framework* JSF. Por fim dois trabalhos que apresentam conceitos e tecnologias que envolvem o *business intelligence* e, mais especificamente, apresentam uma fundamentação teórica sobre a tecnologia abrangendo também os conceitos de *Data Marts* (DM), ferramentas *On Line Analytical Processing* (OLAP) e seus tipos e operações.

Sumário

Estudos de caso-----	6
Análise de Técnicas e Ferramentas para o Desenvolvimento de Interfaces E-commerce Acessíveis para Daltônicos-----	7
Protocolo de Aplicação DDP: Distributed Data Protocol-----	37
Controle de Comprovantes: sistema web desenvolvido com o framework JSF-----	62
Estudo de caso de <i>Business Intellingence</i> : Modelando um Data Mart Sobre Viagens de Transporte Coletivo-----	82
A Implantação do Processo de <i>Business Intellingence</i> – <i>BI</i> na Empresa Samara Calçados-----	116

ESTUDOS DE CASO

Análise de Técnicas e Ferramentas para o Desenvolvimento de Interfaces E-commerce Acessíveis para Daltônicos

Tamiris Cristina de Souza Gouvêa
Sheyla Natália de Medeiros

RESUMO

Esta pesquisa se fundamenta nos conceitos de inclusão e acessibilidade na web, busca o desenvolvimento consciente de sites como e-commerce acerca dos impactos causados por interfaces despreparadas para o público daltônico. Apresenta técnicas e ferramentas que poderão ser usadas por desenvolvedores front-end e ou designer a criarem sites com interfaces inclusivas. Abordam aspectos relacionados à compreensão da deficiência visual na identificação das cores, acessibilidade como suporte aos usuários daltônicos na web e aspectos relacionados ao e-Commerce. Propõe investigar acerca da acessibilidade em websites de Comércio Eletrônico, cujo objetivo é delinear sua estrutura através da interação com usuários daltônicos. Dentre os aspectos metodológicos, é adotado o levantamento bibliográfico com a realização de uma pesquisa exploratória.

Palavras-chave: Acessibilidade na web. Comércio eletrônico. Usuários daltônicos. Teste de acessibilidade, tecnologias assistivas.

1 INTRODUÇÃO

Este trabalho tem a proposta de demonstrar a necessidade da criação de lojas virtuais com *design* acessível para daltônicos, analisar grandes sites de venda como Amazon, Saraiva e Americanas.com, a fim de avaliar seu *design* cognitivo e identificar as dificuldades de visualização existentes para este público.

Busca-se apresentar técnicas para a melhoria do desenvolvimento gráfico Web de e-commerce considerando as normas da WCAG - *Web Content Accessibility Guidelines* da W3C de 2008, que explicam como tornar o conteúdo Web acessível a pessoas com deficiências.

Segundo Queiroz (2009), os padrões web ganham importância quando especificado para que grupo de pessoa se destina a acessibilidade. Com isto será abordado à dificuldade de interação de Daltônicos ao se depararem com uma interface desproporcional a sua deficiência visual e uma possível solução para a prática de desenvolvimentos assistivos, propondo soluções de programação web *front-end* com acessibilidade, podendo ser explorado tecnologias CSS, HTML e JavaScript, fazendo assim com que estes profissionais utilizem em seus projetos, praticas favoráveis a esta inclusão.

O trabalho será desenvolvido com base em uma pesquisa bibliográfica, através de mecanismos de busca como Google Acadêmico e UFPB digital em busca de publicações e artigos relevantes a este tema, utilizando strings de buscas como: e-commerce, Tecnologias assistivas, interface gráfica acessível, usabilidade em sites e-commerce, geração de consumidores web, tipos de comercio eletrônico, estudo da tipografia, psicologia das cores, importância das cores no marketing digital, Daltonicos,

pesquisas atuais do IBGE acerca de deficientes, normas de desenvolvimento assistivo, lei de acessibilidade, entre outros publicados entre o ano de 2008 e 2018. Será realizada uma pesquisa exploratória através de técnicas de desenvolvimento web com Acessibilidade visual.

As cores afetam como as pessoas interagem com o mundo, no comércio eletrônico usa-se cores para dar mais ênfase aos produtos, para chamar atenção para as propagandas e melhorar as vendas, levando em consideração que uma parte da população mundial (até 10%) possui algum tipo de daltonismo esta pode ser uma pratica ineficiente. A condição física dos daltônicos costuma passar despercebida, pois não apresenta diferença visível para os outros. Mesmo assim, os daltônicos enfrentam dificuldades ao longo da vida (Melo ET al., 2014). Desta forma este trabalho busca demonstrar a necessidade de interfaces assistivas para daltônicos.

1.1 OBJETIVOS

1.1.2 GERAL

Demonstrar a necessidade do desenvolvimento de uma interface gráfica acessível, voltada para e-commerce, considerando a interação entre pessoas portadoras de daltonismo.

1.1.3 ESPECÍFICOS

- Analisar a importância do e-commerce e o impacto dele para o mercado;
- Analisar o panorama atual das TAs voltadas ao público pertencente ao grupo dos três tipos de daltônicos mais comuns: protanomalia, deuteranomalia e tritanomia;
- Demonstrar soluções de TAs atualmente aplicadas no desenvolvimento web para e-commerce;
- Comparar TAs mais utilizadas para inclusão de Daltônicos.

2. HISTÓRICO E EVOLUÇÃO DO COMERCIO DIGITAL

O desenvolvimento de lojas virtuais surgiu no ano de 1979 em consequência do aperfeiçoamento da internet, baseado em videoText. O inventor inglês Michael Aldrich que inventou as compras online, utilizando uma televisão personalizada de 26 polegadas para um computador doméstico (Figura 1), que possuía um sistema de processamento de tempo de transação, através de linha telefônica.

As compras on-line foram inventadas, implementadas e tiveram um rápido sucesso entre o modelo grosseiro e a idéia inicial. Foram poucos meses para produzir hardware e *software* e menos de um ano a partir do lançamento do produto para o primeiro sistema operacional, o primeiro negócio direto do mundo para empresas (B2B) sistema de compras on-line em 1981.

Figura 1: Michael Aldrich



Fonte: www.aldricharchive.com

O comércio eletrônico iniciou em 1995, nos Estados Unidos, com o aparecimento de algumas empresas dentre elas a Amazon.com. Contudo o surgimento deste setor no Brasil só ocorreu cinco anos depois, desde então, as vendas através da internet não pararam de crescer (TOREZANI, 2008).

Ocorreram grandes mudanças na forma de se comprar, e o mercado off-line vem tendo que se adaptar a um mercado on-line cada vez mais sofisticado, fazendo com que empresas que não se adequam a ele, diminuam seu percentual de vendas. Um artigo publicado pela revista *The Economist* em março de 2011, mostrou que naquela época nos Estados Unidos as empresas que expandiram suas vendas a esta tecnologia, tiveram mais de cem bilhões de dólares de faturamento.

As compras através da internet vem-se expandindo a cada dia, considerando que hoje grande parte da população está online, seja através de redes sociais, vídeos sobre demanda (VoD), blogs, e outros diversos meios de interação online. A comodidade e facilidade de realizar pesquisas na internet transformam consumidores comuns em aptos compradores, cujo objetivo é pesquisar e avaliar todas as circunstâncias para finalizar uma boa compra considerando valores de mercado e qualidade do material apresentado.

Muitos usuários em potencial buscam o objeto desejado em lojas virtuais e após uma boa pesquisa sobre o mesmo, vão em lojas físicas para comparar valores e até testar a mercadoria, mas com o valor quase sempre imbatível das lojas on-line, retornam ao meio virtual para finalizar a compra. “Um site de e-commerce como loja virtual, tem o objetivo à venda de bens e serviços, portanto uma loja virtual eficaz é aquela que consegue transformar seus visitantes em compradores.” (FELIPINI, 2011, p.16).

A expansão da internet móvel através de dispositivos como *smartphones* e *tabletes* facilitaram ainda mais as vendas nos mercados digitais. De acordo com uma pesquisa apresentada pelo portal R7 de notícias em Janeiro de 2015 – “Estudo revela que brasileiro passa mais de nove horas por dia na internet”, baseado em um estudo publicado pela agência internacional We Are Social, mostrou-se que o Brasil realiza 276 milhões de pesquisa via celular e que 15% dos consultados fizeram alguma compra com o dispositivo móvel, e 22% utilizaram os dispositivos para fazer consultas sobre produtos.

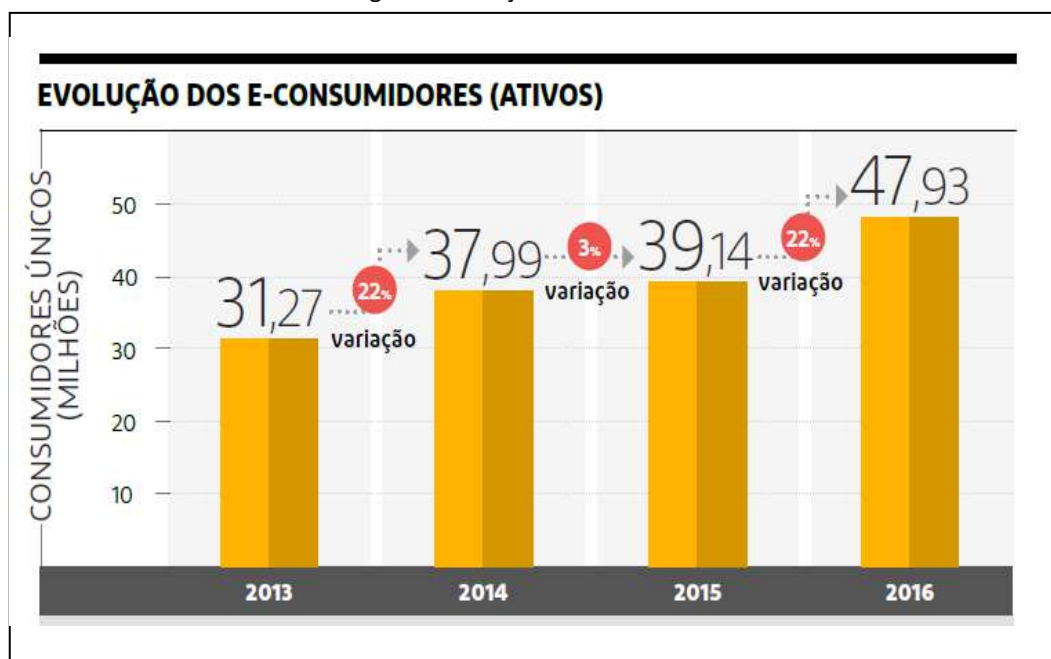
2.1. Impacto das vendas eletrônicas na economia

De acordo com a 35ª e 36ª edição do relatório produzido pela Ebit, o e-commerce brasileiro alcançou em 2016 mais de 44 bilhões de faturamento. Existiu uma alta de 22% (Figura 2) em compras virtuais comparando-as ao ano de 2015, onde aproximadamente 48 milhões de consumidores efetuaram transações online, 21% dessas transações foram realizadas através de dispositivos móveis. O público feminino foi responsável por aproximadamente 51% destas transações enquanto o masculino teve uma média de 48%.

Neste ano as regiões sul e centro-oeste tiveram um maior crescimento no consumo eletrônico enquanto a região sudeste apresentou uma queda de mais ou menos 4% em relação a o ano de 2015, tendo um público consumidor com faixa etária média de 43 anos.

Mesmo com o crescimento do desemprego de 12% no Brasil em 2016, conforme apontado pela PNAD do IBGE, o comércio eletrônico continuou avançando, estimou-se que em 2017 o brasileiro iria faturar mais de 49 bilhões no e-commerce.

Figura 2: Avanço de vendas online



Fonte: IBIT- WEBSHOPPERS 35ª EDIÇÃO

Dentre as expectativas de 2017 surgiu o crescimento das vendas via Marketplace, onde lojas de grande porte abrem suas portas para que pequenos empreendedores possam cadastrar-se e vender em sua plataforma, com objetivo de oferecer maior variedade de produtos ao consumidor e isso vem crescendo abundantemente em 2018, ouve ainda uma grande evolução no m-Commerce fazendo necessário o aprimoramento de seus sites.

A mudança do ranking das categorias mais vendidas em dispositivos móveis é o resultado do investimento e foco de muitas empresas em aprimorar a experiência do consumidor nesses devices. Muitas empresas estão melhorando seus sites responsivos e Apps e ainda oferecendo vantagens

comerciais para alavancar venda nos Smartphones (GUASTI, 2017 apud EBIT, 2017 p. 22).

2.2. Tipos de comercio eletrônico

O que caracteriza o tipo de comércio que uma empresa pratica não é o produto, mas a atividade fim, ou seja, qual o fim a que é destinado à mercadoria. (Nissan, 2014), é possível encontrar diversos tipos de comércio eletrônico como B2B, B2C, C2B, C2C, B2E entre outros, iremos falar mais sobre estes a baixo.

Comércio entre pessoas Jurídicas: *Business to Business* (B2B), nestes são realizadas transações comerciais de empresa para empresa, ou seja, entre duas ou mais pessoas jurídicas, que comercializam seus produtos umas para as outras. Um sistema composto por empresas que utilizam a internet para transacionar seu negócio em busca de atrair todos os participantes de uma cadeia produtiva para um mesmo local, tendo variações como *Business to Employee* (B2C), que ocorre quando uma empresa faz a venda para seus próprios funcionários. *Consumer to Business* (C2B), modelo de negócio onde os consumidores ofertam produtos e serviços às empresas, que contratam e pagam por estes.

Comercio entre pessoas físicas: *Business to Consumer* (B2C) transações realizadas entre pessoa jurídica e pessoa física sendo consumidor final. *Consumer to Consumer* (C2C) desenvolve-se através de usuários particulares na internet onde um consumidor revende a outro consumidor. Operações entre pessoas físicas quando fornecedor é também o consumidor, um dos exemplos mais comuns desta pratica são os leilões.

Neste trabalho procurou-se focar em negociações entre empresas e clientes mais especificamente no conceito de comercio B2C.

2.3. Perfil de e-consumidores

Para o sucesso nas vendas um bom empreendedor deve analisar, e definir qual tipo de público irá atender para com isso especificar qual produto irá comercializar, quais valores deveram ser praticados e quais os locais mais propícios para a divulgação do produto e abertura da loja, pensando desde as cores das paredes até os móveis que irão habitar o interior de sua loja, tendo a intenção de aconchegar e familiarizar o cliente a o ambiente, ganhando sua confiança para impulsioná-lo a adquirir os produtos oferecidos.

As lojas virtuais não ficam fora disto, antes de se criar um e-commerce deve-se estudar cada característica do consumidor a fim de verificar a qual geração ele pertence, sabendo que cada geração possui um perfil próprio que é adquirido ao longo de sua vida.

A maior ignorância dos gestores de negócios centra-se no desconhecimento de quem são seus clientes, por isso, quem deseja atender a esse nicho de mercado deve, inicialmente, buscar compreender o que eles consideram importante e o que os influencia no momento da compra. (UNDERHILL, 1999 apud CERETTA, FROEMMING, 2011, p16)

Um estudo para a criação destes perfis e realizado a cada 10 ou 20 anos onde são analisadas e definidas as gerações de consumidores, podemos citar cinco gerações bem definidas, são elas os Veteranos, Baby Boomers, X, Y, e Z, ambas caracterizadas pelo período em que cresceram, características estas que ajudam a definir como cada

consumidor irá reagir diante de aspectos específicos. Existe uma complexidade em analisar estes perfis uma vez que envolve especialidades e comportamentos peculiares dos clientes.

Para personalizar a visão de um consumidor é necessário entender o que ele compra, onde, quando e porque compra. Com base nisto iremos detalhar um pouco mais cada um destes perfis explorando matérias como a de Fábio Turci (2010) publicada no G1 e Sylvia de Sá (2010), publicada na revista EXAME.

2.3.1. Geração de Veteranos e Baby Bombers

A geração dos veteranos é composta por pessoas nascidas antes e durante a II Guerra Mundial até meados de 1945, com aproximadamente 65 anos de idade, influenciados pelo muro de Berlim e uma grande guerra, possuem como característica o respeito pela hierarquia sendo bem disciplinados, obedientes e conservadores geralmente compram o necessário sem exageros.

Geração de Baby Bombers nascidos no pós-guerra entre 1945 e 1965 foram jovens ativos nos movimentos estudantis nas décadas de 60 e 70, que vivenciaram a ditadura militar no Brasil estes consumidores valorizam status, gostam de gastar e investem em bens materiais como belas jóias, carros e casas, buscam crescimento profissional e intelectual.

2.3.2. Geração X e Y

Pessoas que nasceram entre 1961 e 1980 consideradas da geração X possuem algumas resistências sobre as tecnologias, são apegados a títulos e cargos, mas visão a qualidade de vida e a liberdade no trabalho procurando um equilíbrio entre elas. Viveu o período da guerra fria, esta foi a primeira geração a dominar os computadores, são grandes consumidores de livros e filmes, tendo uma resistência nas compras online, mas com o desenvolvimento dos serviços e tecnologias de hoje, estão mais confiantes neste mercado.

Já a geração da Internet nascida entre 1980 e 2000, denominada geração Y, é amante da tecnologia e busca a intensidade em cada experiência vivida, estes executam multitarefas e são imediatistas, uma geração de diversidades, ligada em artes, filmes e cinema.

Marcados pela necessidade de reconhecimento por tudo que fazem, também, são flexíveis, criativos e questionadores. Habitados com o mundo dos videogames, agora, eles estão desembarcando, competitivos e impacientes, no ambiente empresarial. A Geração Y tem pressa. (OULIVEIRA, 2010 apud LINS, 2011 p.1).

2.3.3. Geração Z

Nascidos após o ano 2001 junto à tecnologia são mais exigentes em relação a suas escolhas, não sabem o que é viver em uma época sem celulares, tabletes e outras tecnologias, preocupam-se com a sustentabilidade. Estes são grandes consumidores ligados a tendências, valorizando marcas e produtos inovadores. É um grupo que costuma ter sua própria renda, seja por mesada dada por seus familiares ou por seu próprio mérito, trabalhando como jovens aprendizes em grandes e pequenas empresas.

São grandes influenciadores na decisão de compra dos seus pais ou parentes mais velhos.

Em um estudo realizado com jovens de nove países diferentes, confirma o prazer que possuem pelas compras e destaca que, no Brasil, sete, em cada dez, jovens admitem gostar de realizar compras, enquanto que quatro, em cada dez, afirmam ter grande interesse pelo assunto. O resultado da pesquisa evidenciou que os brasileiros estão em primeiro lugar no ranking de consumo. (RUBENS 2003 apud CERETTA, FROEMMING, 2011, p18).

Uma geração independente que busca a liberdade, e gosta de customizar tudo a sua volta, este sempre conectado a redes sociais a procura de informações claras e contato com os amigos, deseja em seu local de trabalho e educacional diversão e entretenimento. No mercado comparam todos os preços e levam em consideração a reputação dos produtos publicada por outros consumidores em suas redes.

3. INTERFACES GRÁFICAS NA WEB

A interface gráfica surgiu na década de 1980 substituindo as interfaces de texto, para tornar mais intuitiva e fácil a utilização das máquinas, pois a forma de interação entre o homem e o computador sempre foi uma das preocupações na indústria da informática. Utilizar-se de recursos gráficos para localizar a informação e favorece a memória humana.

O nosso universo está repleto de imagens. O nosso pensar passa pelas imagens. O nosso sentir não as ignora. O nosso agir habituou-se a lidar com elas. Uns acham-nas necessárias, outros excessivas, outros ainda supérfluas (ABRANTES, 1999). Os homens conseguem reter mais as imagens do que os textos.

Interfaces podem inspirar sentimentos que determinarão como um usuário se comportará em um sistema operacional. Baseado em estudos de Lordelo (2007) 86% das pessoas optam em abandonar o uso de um programa devido sua interface, de acordo com o filósofo Aristóteles no Livro II da Retórica “As emoções são aqueles sentimentos que nos mudam de uma forma capaz de afetar o raciocínio e que são acompanhados de prazer ou dor.”, é por meio das interfaces que os usuários acessam a função das aplicações.

Interface é a zona de comunicação em que se realiza a interação entre o usuário e o programa. Nela estão contidos os tipos de mensagens compreensíveis pelos usuários (verbais, icônicas, pictóricas ou sonoras) e pelo programa (verbais, gráficas, sinais elétricos e outras), os dispositivos de entrada e saída de dados que estão disponíveis para a troca de mensagens (teclado, mouse, tela do monitor, microfone) e para as zonas de comunicação habilitadas em cada dispositivo (as teclas no teclado, os menus no monitor, barras de tarefa, área de trabalho) (GALVIS, 1992 apud LORDELO, 2007, p. 31).

O intuito de desenvolver uma interface é permitir e auxiliar o usuário a cumprir sua tarefa desejada com a maior facilidade e o menor atrito possível, proporcionando uma sensação agradável e de bem estar ao usuário. Para o desenvolvimento de interfaces Web é primordial a criação de um *Style Guide*, ou seja, um conjunto de normas objetivadas a proporcionar um desenvolvimento uniforme na formatação e no estilo do documento, para isto deve ser considerado o Layout de composição, onde se aplica o

conceito de disposição dos elementos em uma página, a tipografia, a paleta de cores, as imagens e as Guidelines da marca.

Para o desenvolvimento de um site é preciso se preocupar com todos os tipos de usuários, pode-se usar conceitos de *Responsive Web Design*, que nada mais é do que layouts que se adaptam de acordo com as características do dispositivo utilizado pelo usuário, porém o Web Designers deve salientar a fluidez da diversidade na web.

É preciso estudar o público alvo na hora de organizar o planejamento de um Layout, para buscar as características específicas que iram fazer com que seu usuário estabeleça uma boa conexão com o conteúdo em busca da melhor interação entre ele e a página, sendo necessário estudar como estruturar os conteúdos dentro da mesma em busca da melhor comunicação entre o conteúdo e o usuário, afinal quando as pessoas procuram informações e se deparam com interfaces fora do seu contexto, tendem a ignorar o conteúdo ou acabam não adquirindo a informação disposta a ela, por isso é necessário se preocupar com usabilidade da interface projetada.

3.1. Tipografia

Para o processo de composição e criação de um texto, é necessário usar-se da tipografia, onde cada letra precisa ser alinhada e detalhada para formatar seu estilo, existem diversas formas de letras desde as neutras, informais, atemporais, entre outras.

À organização da tipografia é feita através das fontes, estas que variam desde as letras distintas e claras que são geralmente usadas em grandes textos, até as mais dramáticas que geralmente são aplicadas em publicidade e propaganda. Usamos esta em quase tudo, podemos encontrá-las nas revistas, jornais, livros, em tudo que é relatado através de texto, usa-se fontes tipográficas, o uso dos diversos tipos caracterizar a informação passada ao leitor.

A tipografia é o meio pelo qual é dada uma forma visual para uma idéia escrita. Devido ao volume e à variedade de fontes disponíveis, a seleção dos componentes desta forma visual pode afetar drasticamente a legibilidade da idéia e os sentimentos do leitor em relação a ela. A tipografia é um dos elementos que mais influencia o caráter e a qualidade emocional de um projeto. Ela pode produzir um efeito neutro ou despertar paixões, simbolizar movimentos artísticos, políticos ou filosóficos, ou ainda expressar a personalidade de um indivíduo ou organização. (AMBROSE; HARRIS, 2011, p.6)

A maneira como seu texto é composto e disponibilizado, afeta o modo no qual a mensagem será lida, tipografia corresponde à disponibilidade das letras em um projeto para intensificar o significado das palavras, existem diversos tipos de fontes que são definidas pelos autores Ambrose e Harris como:

Um Tipo é um conjunto de caracteres, letras, números, símbolos, pontuação (e assim por diante) que tem um design comum e distinto. Uma fonte é o meio físico utilizado para criar o tipo, seja ele código de computador, fotolito, metal ou gravação em madeira. (AMBROSE; HARRIS, 2011, p.17)

O designer tipográfico é capaz de aproximar as empresas ao consumidor, através de uma linguagem comum entre eles, a tipografia possui valores funcionais e simbólicos que personificam a informação, capazes de alcançar o sentimento dos consumidores. A experiência do usuário é dada através da interface do produto, por tanto um bom designer deve-se preocupar com a forma visual da pagina, é preciso ter o foco no

usuário em todas as etapas de desenvolvimento de um projeto, a maneira como são formadas e dispostas às letras, alteram nossa forma de percepção sobre o que elas retratam conforme demonstrado a figura 3.

Figura 3 - Forma e apresentação de letras (Tipografia)



Fonte: Tipografia (AMBROSE; HARRIS,2011)

Existem normas como a da ABNT que descrevem como um texto deve ser escrito, para o desenvolvimento web pode-se basear tanto em normas de escrita como em experiências de usuários para proporcionar a melhor visibilidade de um texto, em links, por exemplo, quando colocados em uma página web, usam-se geralmente textos sublinhados e com uma cor diferenciada da cor do texto original e quando clicado, mais uma vez modifica-se esta cor, porém títulos e subtítulos seguem os princípios das normas formais de escrita, sendo sempre colocados em evidência, através de tamanhos diferenciados e ou fontes em negrito, com o intuito de permitir que o texto fique mais robusto aos olhos do leitor.

O negrito faz parte da família Helvetica Neue, que possui diversas variações de escrita, como Itálico, condensado, claro ou fino e outras mais, que são usadas para esclarecer melhor o seu texto aos olhos do usuário de acordo com Ambrose Hares (2011) essas variações dão ao designer mais flexibilidade para definir um formato interessante e útil sem deixar as características do tipo.

3.2. Definição de Cores

Cores são reflexos das luzes captadas pelos olhos e recebidas pelo cérebro, sensações subjetivas produzidas pelo mesmo, são radiações eletromagnéticas que podem ser medidas pelo comprimento de suas ondas, onde cada comprimento representa uma cor em particular.

Por meio de um experimento realizado por Isaac Newton em 1074, foi possível observar que com a união das cores cria-se a sensação da cor branca e a ausência de todas elas, leva-se a cor preta, na terceira edição do livro "A Cor Como Informação" de

Guimarães (2000), baseando-se nos estudos de Isaac Newton, relatou-se que “são sete as cores fornecidas pela luz branca: vermelho, alaranjado, amarelo, verde, azul, anil e violeta”.

Para trabalhar com as cores, dividem-se as mesmas em cores luz RGB, pigmento CMYK, que são objetos das cores luz entre outros modelos cromáticos. Com base na pesquisa de Pedrassolli e Almeida (2014), seguindo as teorias de Young-Helmholtz (Farina et al., 2006) também encontrada como Teoria Tricromática, “existem no olho humano três cones receptores (células especializadas que compõem a retina) sensíveis a luz, cada um relativo a percepção de uma cor”.

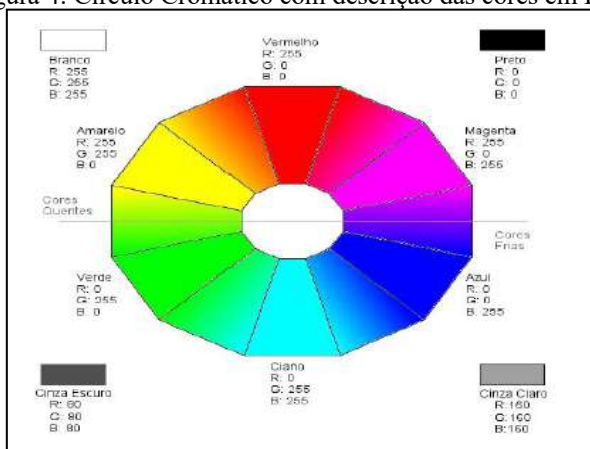
Com isto podemos relatar que o homem tem percepção das cores através dos cones, encontrados dentro dos olhos, eles identificam as cores primárias (RGB) e quaisquer outras cores podem ser produzidas através destas, utilizando-se do conceito de ausência ou presença de luz “A cor não tem existência material: é apenas sensação produzida por certas organizações nervosas sob a ação da luz – mais precisamente, é a sensação provocada pela ação da luz sobre o órgão da visão” (SANDRONI, 2011).

A percepção das cores ocorre de forma fisiológica que se modifica de pessoa para pessoa dependendo também de fatores psicológicos, culturais ou sociais distintos. Na Web a cor é um dos primeiros elementos da interface a ser observada pelo usuário, a mesma em conjunto com os outros componentes da página compõem a linguagem visual do site.

Aplicar a teoria das cores nestas interfaces pode melhorar a visão do usuário, otimizando o tempo de resposta e evitando a decepção do usuário em contato com a página, diminuindo assim a desistência na hora da navegação. Na linguagem HTML usadas na programação web, são adotadas as cores-luz, formada pelo conjunto de cores vermelho, verde e azul, “em uma variação de 0 a 255 para cada cor primária.” (PEDRASSOLLI; NERIS, 2014), Figura 4.

As escolhas de cores para projetos Web podem se basear em uma metodologia de aplicação de cores através do círculo cromático. Combinações básicas são feitas a partir de uma cor e suas variações até o cinza ou a combinação com uma cor neutra, como marrom, que está fora do círculo de cores e tenha luminosidade diferente. Combinações entre cores complementares ou opostas são comumente utilizadas em tons quebrados, onde a variação das cores passa por diferentes saturações, ou em tons puros. (PEDRASSOLLI; NERIS, 2014, p. 3)

Figura 4: Círculo Cromático com descrição das cores em RGB



Fonte: Projeto acadêmico de PEDRASSOLLI e NERIS (2014)

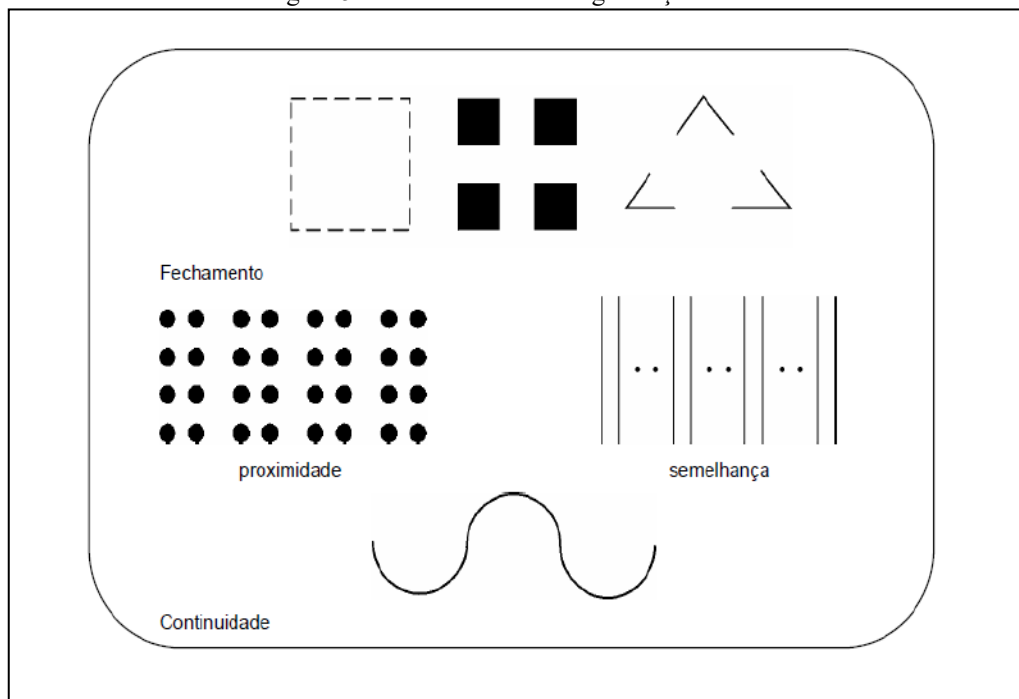
As cores possuem diversas características próprias que variam de autor para autor, basicamente trabalham-se com a matriz, a temperatura e a saturação para adequar as mesmas em um projeto.

3.3. Estruturação de Páginas

Para facilitar o acesso do usuário ao conteúdo apresentado é necessário saber manipular os elementos visuais para que seja percebido e entendido pelo mesmo. Torna-se necessário sua organização em grupos e a distribuição do conteúdo em diferentes níveis, que serão visualizados e processados pelo cérebro que terá como percepção todo o conteúdo em forma global e unificada, não se visualiza partes isoladas tal qual o primeiro contato de visualização através da retina.

Inicialmente são percebidas as forças de unificação e segregação e em seguida são identificadas forças internas que orientam os elementos que compõe a unidade, conhecidos com Leis de Organização Visual conforme relatado por Gabriela Mager (2004), leis estas que defendem a percepção das informações através dos fenômenos: Fechamento, onde tende-se psicologicamente a estabelecer ligações e unir intervalos entre o conteúdo interpretado; Continuidade que se tem a impressão de que as imagens são sucedidas umas das outras através da organização perceptiva da forma de diferentes modos coerentes sem quebras em sua trajetória e ou fluidez visual; Proximidade onde elementos óticos que estejam próximos uns dos outros tem a tendência de serem visualizados juntos, ou seja criam unidades dentro de um todo; Semelhança, elementos que por sua vez possuem qualidades em comum, como igualdade em formato ou cor, estes tendem-se a se agrupar com unidades semelhantes construindo uma única unidade (Figura 5.)

Figura 5: Modelo da Lei de Organização Visual



Fonte: Dissertação de Gabriela B Mager (2004,apud Gomes, 2000, p. 21-23)

A partir da disposição das informações graficamente, pode-se definir a hierarquia de posição das informações e a relação entre elas, melhorar o impacto de percepção em pontos desejados, direcionar a transação entre as unidades de sentido e

traçar o circuito da leitura, esta diferenciação visual no layout da interface, em que a forma tipográfica, as cores e o espaço são manipulados, podem por objetivo, guiar o olhar do usuário na intenção de manipular sua percepção (MAGER, 2004 p. 28).

As informações dispostas em sua página virtual, devem ser colocadas de forma equilibrada para compor uma visualização saudável, precisa-se tomar os devidos cuidados em relação ao fundo escolhido para envolver o projeto, o contraste dos dados e o posicionamento das informações considerando que informações próximas umas das outras, tem-se a impressão de um grupo de dados, já as mesmas dispostas distantes, compreende-se que não fazem parte de um mesmo grupo.

Esta providência é necessária para se ter uma página ergonomicamente acessível ao usuário, proporcionando o bem estar em toda a sua navegação levando o mesmo a passar mais tempo em seu site.

A imagem gráfica é considerada um elemento da comunicação humana. Se a interface gráfica for organizada por princípios ergonômicos e de design gráfico, ela será mais eficiente e atrativa, resultando em um usuário mais motivado a ler a informação e a entendê-la mais facilmente. Além da percepção clara da informação, a interface deve levar em consideração o uso que será feito dela. Para tanto, destaca-se a usabilidade. ((MAGER, 2004 p. 30)

4 ACESSIBILIDADE

Acerca do termo acessibilidade, denota-se que esta é a qualidade do que é acessível, do que tem acesso, deve expressar a facilidade, possibilidade na aquisição, na aproximação daquilo que se procura. Deve-se possibilitar o alcance do indivíduo com deficiência ou mobilidade reduzida de forma que aja segurança e autonomia do mesmo. Para o desenvolvimento de tecnologias é necessário planejar uma estrutura acessível.

Art. 2º Considera-se pessoa com deficiência aquela que tem impedimento de longo prazo de natureza física, mental, intelectual ou sensorial, o qual, em interação com uma ou mais barreiras, pode obstruir sua participação plena e efetiva na sociedade em igualdade de condições com as demais pessoas. (Rouseff, 2015 ,Lei Nº 13.146, p. 1 Art. 2)

De acordo com a lei Brasileira de Inclusão da Deficiência Nº 13.146, é necessário assegurar e promover igualmente o exercício dos direitos e da liberdade fundamental por pessoas com deficiência, buscando sua inclusão social e cidadania. Torna-se necessário fornecer possibilidades e condições de alcance para utilização com segurança e autonomia, de espaços, transportes, comunicação, inclusive sistemas de tecnologias bem como de outros serviços e instalações abertas ao público.

4.1 Tecnologias Assistivas

O termo Tecnologia assistiva é usado para identificar todos os serviços e recursos que proporcionam ou ampliam as habilidades funcionais de pessoas com deficiência tendo como conseqüências a vida independente e inclusão, este é o elemento crucial para a promoção dos direitos humanos. A cada dia vem-se ampliando o interesse dos brasileiros sobre esta temática, podemos observar este através das inúmeras exposições em feiras sobre esta tecnologia, onde algumas sucedem edições anuais. Em diversas regiões do Brasil existem grupos de pesquisas acadêmicas e de terceiro setor voltados a esta temática.

"Tecnologia Assistiva é uma área do conhecimento, de característica interdisciplinar, que engloba produtos, recursos, metodologias, estratégias, práticas e serviços que objetivam promover a funcionalidade, relacionada à atividade e participação de pessoas com deficiência, incapacidades ou mobilidade reduzida, visando sua autonomia, independência, qualidade de vida e inclusão social" (CORDE 2009, p. 9 apud CORDE 2008)

Podemos considerar como produtos assistivos todos os equipamentos, produtos, instrumentos ou tecnologias que são projetadas ou adaptadas para melhorar e facilitar a independência na utilização destes, assegurando que uma pessoa portadora de deficiência ou mobilidade reduzida, possa utilizar estes recursos que compensam ou neutralizam a deficiência ou incapacidade.

Desenvolver soluções tecnológicas baseados em TA coopera para que o usuário usufrua de uma tecnologia que realmente atenda suas expectativas e necessidades. Conforme demonstrado pela Coordenadoria Nacional para Integração da Pessoa Portadora de Deficiência, para se definir a TA adotam-se três referências: ISSO 999, Classificação HEART e a Classificação Nacional de Tecnologia Assistiva, do Instituto Nacional de Pesquisas em Deficiências e Reabilitação, dos Programas da Secretaria de Educação Especial, Departamento de Educação dos Estados Unidos, que levam a concluir que não existe uma forma única de definição para a TA e que as várias classificações são aplicadas de acordo com os objetivos de catalogação de recursos, ensino, trocas de informação, organização de serviços de aconselhamento e concessão. (CORDE, 2009, p.25.).

Existem duas definições de modelos de prestação de serviço em TA, segundo o CORDE baseado em dados do CSUN, 2006, que são denominados como paradigmas do déficit Individual tendo como foco as pessoas com deficiência e o paradigma tecnológico/ecológico focado nos sistemas de recursos de TA, que tomam como estratégia o desenvolvimento de tecnologias e sistemas direcionados as necessidades do consumidor com algum tipo de dificuldade funcional ou deficiência.

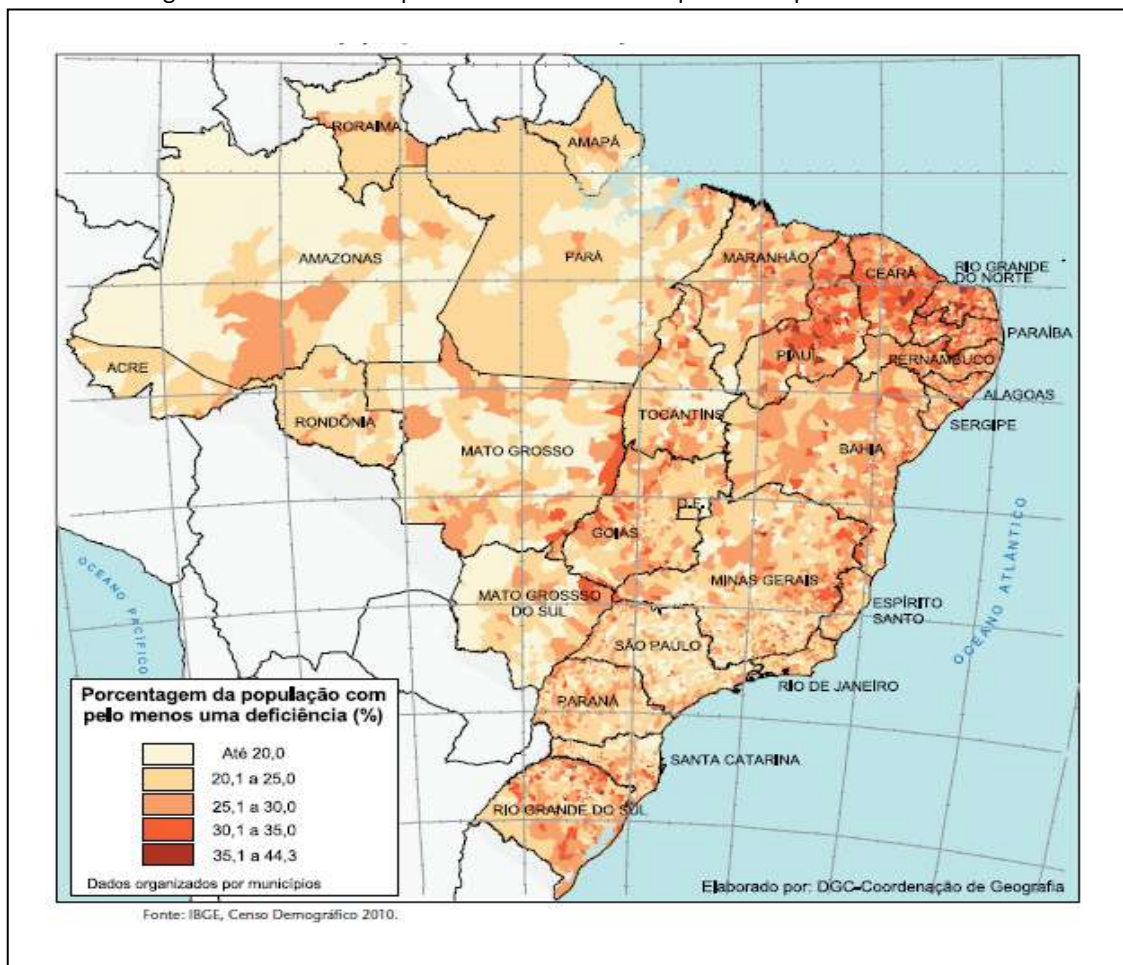
O sucesso de qualquer projeto, prescrição ou utilização de TA depende da ação integrada e complementar de diversas áreas do conhecimento com um objetivo último comum, que é a satisfação das necessidades do usuário com deficiência, em todas as esferas da sua atuação pessoal, doméstica e comunitária. Neste contexto, o usuário deve ser tratado e incentivado a ser um consumidor consciente. (CORDE, 2009, p.28.)

4.2. Índice de pessoas com alguma deficiência no Brasil

Na decretação elaborada por Italo Guimarães e Marckson Souza a fim de defender a necessidade do desenvolvimento Web para usuários cegos, em 2015, foi detalhado que de acordo com a última cartilha do CENSO 2010, apresentou-se um panorama a cerca da população brasileira com deficiência, destacando que mais de 45 milhões de brasileiros, cerca de 23,9%, são portadores de algum tipo de deficiência, onde 26,5% são mulheres e 21,2% homens.

Ainda com base nos dados do IBGE demonstrados no CENSO 2010, denota-se que 38.473.702 dessas pessoas são encontradas em áreas urbanas e 7.132.347 em áreas rurais, destacando a região Nordeste que concentra o maior percentual de habitantes com uma das deficiências investigadas conforme figura 6.

Figura 6: Percentual de pessoas com deficiências por municípios - Brasil - 2010



Fonte: IBGE Censo Demográfico 2010

Até este momento observamos que o desenvolvimento assistivo é uma prática não só necessária, mas também obrigatória defendida por leis e diretrizes. Podemos considerar, que um local onde uma pessoa portadora de deficiência ou mobilidade reduzida, não consegue ter acesso por falta do desenvolvimento destas práticas e tecnologias é um local deficiente e não que a pessoa que tenta acessar estes ambientes e não consegue, seja uma pessoa incapaz. Com tudo uma grande maioria de desenvolvedores web não costumam implementar ambientes adaptados, deixando as aplicações web bem distante daquilo que se considera ideal em termos de TA.

Pensando neste desenvolvimento devemos considerar que muitas pessoas possuem dificuldades não tão notáveis como paraplégicos, cegos, surdos, mudos, entre outros e que dentro dos milhares de pessoas portadoras de deficiência encontram-se os Daltônicos, pessoas estas que enfrentam uma grande dificuldade para fazer a distinção de cores. Para esta pesquisa iremos nós aprofundar nos estudos voltados a estes usuários, levando em consideração que aproximadamente 10% da população possui algum tipo de daltonismo onde 98% destes são do sexo masculino conforme figura 7.

Figura 7: Dados da pesquisa de Miguel Neiva acerca dos Daltônicos



Fonte: Percentual TEDxESPM - Miguel Neiva - ColorADD

Como vimos anteriormente, para se desenvolver uma loja virtual de sucesso, é necessário alinhar as cores, as letras e a posição dos ícones em sua interface, tornando-se necessário a avaliação e interação dos usuários para a criação de e-commerce adaptável a este público que representa uma boa parte da nossa sociedade.

Algumas pessoas têm dificuldades em identificar cores, por isso são popularmente chamados de daltônicos. Entre 6% e 10% dos homens e 0,4% a 0,7% das mulheres são daltônicos no mundo. Apesar de não aparentar, daltônicos relatam dificuldades em diferentes fases da vida. (BRUNI, CRUZ 2016; GORDON, 1998; MELO 2014; apud ANDRADE, PINTO, SILVA, 2017, p.2.)

A ausência de pesquisas e práticas de acessibilidade web, diminui a relação de negócio com potenciais consumidores que tem alguma deficiência visual na interpretação de cores. Desta forma, comerciantes que atuam ou pretendem atuar na web devem adequar suas páginas a fim de atender a necessidade de usuários daltônicos gerando laços de confiança com este público atraindo-os para efetuar compras em seu e-commerce.

5. DALTÔNICOS

O problema de visão na hora da identificação de cores popularmente conhecido como daltonismo, pode ser congênito resultante de uma alteração genética ou decorrente de doenças sistêmicas ou oculares é cientificamente conhecido como Discromatopsia em referência ao químico John Dalton, 1766-1844, que tinha protanopia (um tipo de discromatopsia) e foi o primeiro cientista a estudar o assunto (BRUNI; CRUZ, 2006 apud MELO; GALON; FONTENELLA, 2014).

Como já observado neste trabalho, a identificação das cores é um fenômeno que se relaciona com células fotossensíveis em especial os cones que possuem tipos específicos de fotopsinas, proteínas responsáveis pela conversão de sinais, ondas luminosas, em elétricos que são levados até o cortex cerebral através do nervo óptico onde a visão combinada das cores é interpretada. O material genético responsável pela codificação das fotopsinas pertence ao mesmo gene das proteínas receptoras do odor e do paladar (JACOBS, 2009 apud MELO; GALON; FONTENELLA, 2014).

Quando os cones não conseguem diferenciar ou identificar estímulos de luzes correspondentes a identificação das cores verde, vermelho ou azul, as pessoas têm dificuldades de enxergar estas e suas derivações.

Existem tipos específicos de daltônicos, alguns apresentam dificuldades em identificar e derivar a cor vermelha, denominados como Protanomalia; verde, diagnosticados como Deuteranomalia; azul, identificados como Tritonomalia e à ainda, pessoas nas quais os cones são incapazes de identificar todas estas cores, devido a não identificação do comprimento das ondas, levando estas à incapacidade de visualizar qualquer tipo destas cores, são pessoas que observam o mundo em tons de cinza, preto e branco, uma condição mais rara que é conhecida como acromatismo ou acromacia.

Mesmo com todos estes diagnósticos e grupos de pessoas daltônicas existentes no mundo, esta dificuldade costuma passar despercebida diante da população, por não ser uma deficiência física visível a todos levando estes usuários a enfrentarem grandes barreiras diante das tecnologias existentes.

5.1 Desenvolvimento acessível para Daltonicos

Vimos até aqui que o desenvolvimento assistivo é necessário e que para desenvolver tecnologias web em especial neste caso, páginas e-commerce, é necessário preocupar-se com todos os tipos de usuários em potencial, pois a intenção de um e-commerce e a interação com o usuário, proporcionando confiança e segurança a fim de vender produtos e serviços ao mesmo. Para esta prática de desenvolvimento, pode-se utilizar das regras dotadas pela principal organização de padronizações web, a W3C utilizando-se das diretrizes da WCAG onde se reúne uma série de recomendações para acessibilidade na web.

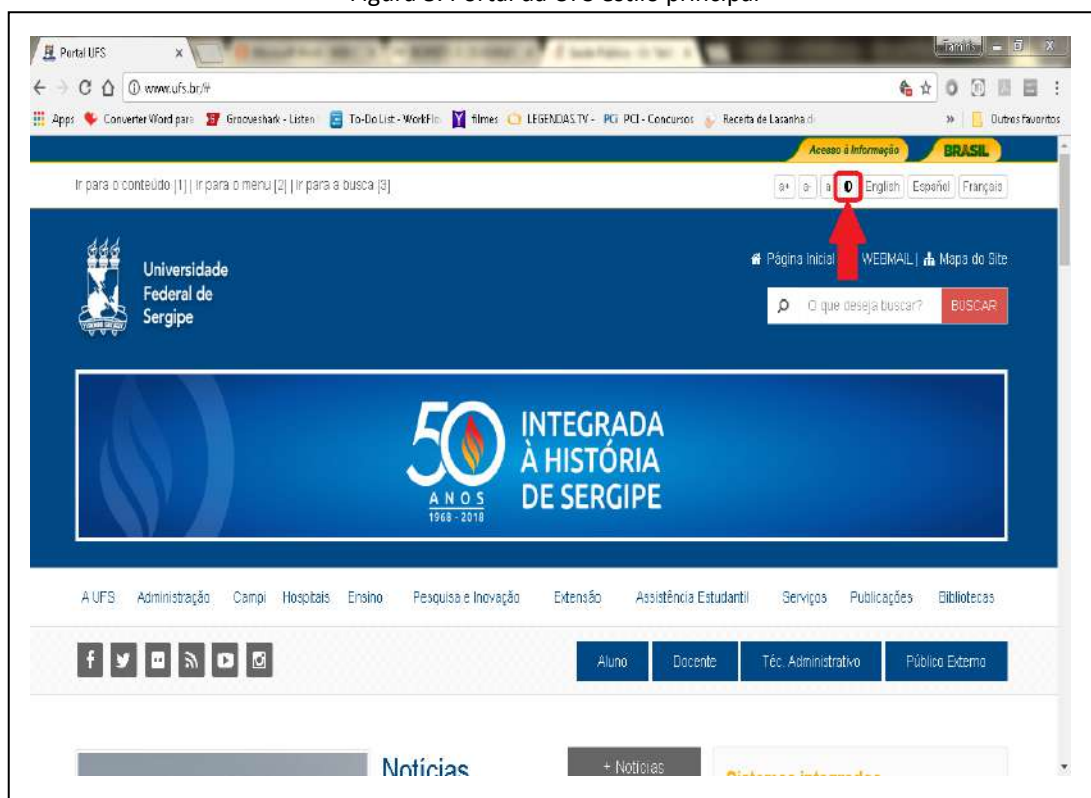
As tecnologias de comunicação e informação estão proporcionando a inserção de pessoas com necessidades especiais no acesso a web. Assim, “Projetar interfaces fáceis de usar significa projetá-las de acordo com o modelo mental de seus usuários considerando, também, aspectos de suas possíveis limitações físicas” (PAIVA, 2000 apud SILVIA, SILVEIRA, OLIVEIRA, 2012, p. 85).

De acordo com os paradigmas da W3C, homens e software específicos avaliam os sites no intuito de validá-los, o meio mais correto para a implementação de sites é começar a programar incluindo as indicações impostas por esta organização. Uma forma de desenvolvimento pensada para usuários daltônicos e a criação de folhas de estilos Css alternativas, versões opcionais do site que disponibilize um contraste diferenciado da página visando à inclusão de daltônicos no acesso das páginas.

Pode ser observado um exemplo desta pratica no portal da Uiversidade Federal de Sergipe, onde ao clicar no ícone de contraste localizado na parte superior da pagina, imediatamente a mudança do estilo da pagina é observada, praticamente todo site fica com um contraste preto e branco (figura 8 e 9).

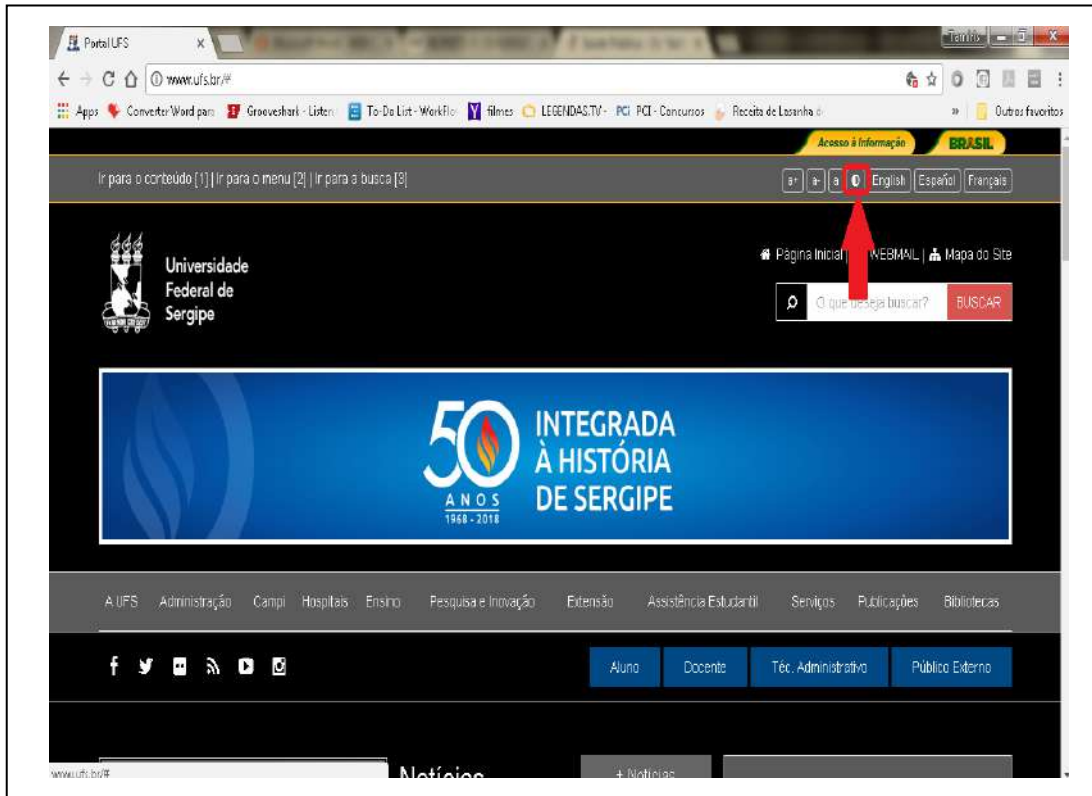
Este tipo de estilo não atende todas as necessidades dos daltônicos e do desenvolvedor, pois ele não é aplicado nas imagens inseridas no site e acaba limitando o desenvolvedor na criação de suas interfaces, levando-o a minimizar sua criatividade na hora de construir o designer, podendo fazer com que seu projeto não atinja o seu publico de uma forma que o impulse a navegar e comprar em sua loja, porem podemos considerar este um grande passo a se desenvolver sobre esta opção de pagina adjacente pois possibilita maior interação deste publico e a interface apresentada.

Figura 8: Portal da UFS estilo principal



Fonte: <http://www.ufs.br>

Figura 9: Portal da UFS estilo adaptado



Fonte: <http://www.ufs.br>

Com base nesta técnica usada na universidade Federal de Sergipe, tendo como objetivo a inclusão, procurou-se avaliar lojas e-commerce como: Americanas, Amazon, Magazine Luiza, Submarino, Walmart e Saraiva, lojas estas classificadas pela EBIT como as melhores lojas e as mais queridas, pelo público em 2018, buscando técnicas semelhantes ou melhores, tendo o mesmo objetivo desta faculdade de Sergipe, proporcionar inclusão e igualdade.

Porem em nenhuma destas destacadas a cima pode-se encontrar um estilo de pagina adaptado para usuários daltônicos, levando a concluir que este tipo de desenvolvimento deixa a desejar nas atuais plataformas de venda do mercado. Será demonstrado mais a frente neste trabalho, o comportamento visual de algumas destas lojas através de ferramentas e técnicas que podem possibilitar o desenvolvedor front-end a criar lojas inclusivas para Daltonicos.

6 SIMULADORES PARA AUXILIO AO DESENVOLVEDOR NÃO DALTÔNICO.

Aprofundando-se sobre tipos de desenvolvimentos assistivos para daltônicos, foi possível observar algumas ferramentas que possibilitam simular a visão de um daltônico, para facilitar o desenvolvedor não daltônico a entender a dificuldade visual destes usuários. Destacam-se algumas destas ferramentas no artigo: Simuladores de Tecnologias Assistivas para Daltônicos, criado por Gilmar Andrade, Joseh Pinto e Bruno Silva em 2017, estas ferramentas foram analisadas e apontadas com índice de qualidade entre bom, regular e ruim (figura 10.).

Figura 10: Simuladores de visão para usuários não Daltonicos

Simulador	Plataforma	Qualidade	Restrições	Diferencial
Simuladores de imagem (estático - interface)				
Colour Blindness Simulator	sistema web	Ruim	Arquivo JPEG de até 100KB. Resultado em baixa resolução. Apenas um tipo de daltonismo por envio de arquivo.	
Chromatic Vision Simulator	sistema web	Regular		Facilita a comparação lado a lado de simulações dos três tipos de daltonismo com apenas um envio de imagem.
Vischeck	sistema web	Boa	Apenas um tipo de daltonismo por envio de arquivo.	Aceita vários formatos e tamanhos de arquivo.
Simuladores de tela (estático - interface)				
Color Oracle	sistema desktop	Boa	Não é possível interagir com o sistema durante a simulação	Altera todas as cores.
Simulador de sites web (dinâmico - interação e interface)				
Spectrum	complemento de navegador	Boa	Funciona apenas no Google Chrome	Altera todas as cores. É possível interagir durante a simulação.
NoCoffe Vision Simulator	complemento de navegador	Boa	Funciona apenas no Google Chrome	Altera todas as cores. É possível interagir durante a simulação. Também simula outras alterações visuais.
Color blinding	complemento de navegador	Boa	Funciona apenas no Google Chrome	Altera todas as cores. É possível interagir durante a simulação.
I want to see like the colour blind	complemento de navegador	Boa	Funciona apenas no Google Chrome	Altera todas as cores. É possível interagir durante a simulação.
ColorFilter	sistema web	Ruim	Desconfigura <i>layout</i> da página. Simulação desaparece ao mudar página.	
Simulador via câmera de dispositivos miveis (dinâmico - interação)				
Chromatic Vision Simulator	aplicativo móvel	Boa	Pode apresentar atraso na simulação em alta qualidade.	Divide a tela em até 4 compartimentos e compara todos os tipos de daltonismo. Permite zoom e salva imagem da tela.
Daltonizer	aplicativo móvel	Boa	Pode apresentar atraso na simulação em alta qualidade. Não compara os tipos de daltonismo.	Divide a tela em 2 compartimentos. Permite congelar imagem e ligar/desligar flash da câmera.

Fonte: Artigo: Simuladores de Tecnologias Assistivas para Daltônicos.

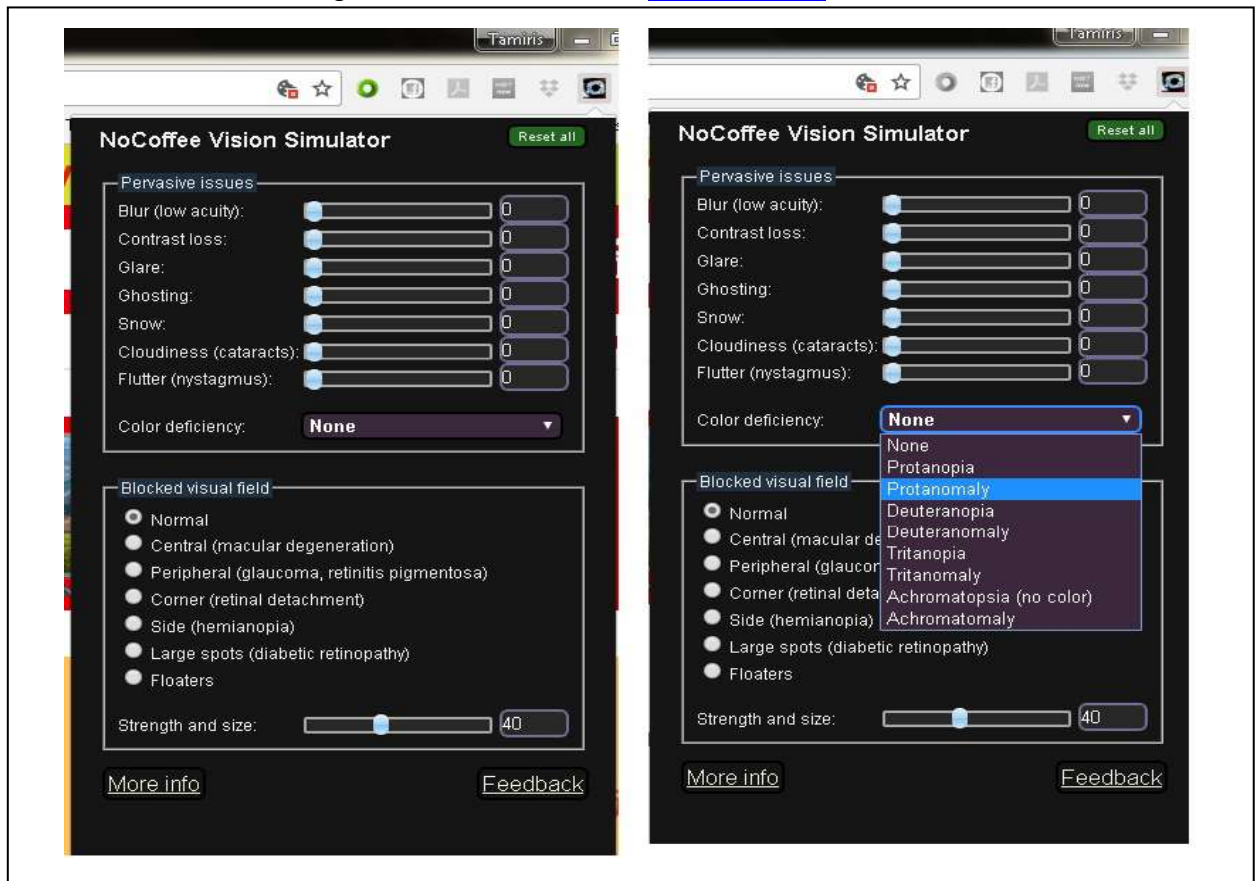
Neste trabalho tomaremos como foco o NoCoffe Vision Simulator, um simulador dinâmico voltado para sites web e o Color Oracle, que possibilita a simulação de telas estáticas, devido a estes terem sido apresentados no artigo como simuladores de boa qualidade.

6.1 Simulador de site Web

O Simulador [NoCoffe Vision](#) é uma extensão gratuita, que possibilita o entendimento de diversos problemas visuais leves ou extremos, ela pode ser adicionada ao seu navegador para criar simulações visuais diversão em todo site, assim que instalado o ícone da extensão será percebido na barra de ferramentas do navegador.

Depois de adicionado esta extensão, o desenvolvedor pode inspecionar a sua aplicação, a navegação simulada pode ser iniciada em qualquer momento. Clicando no ícone da extensão, será possível ter acesso a o menu demonstrado na figura 11, que possibilita escolher que tipo de dificuldades visuais deseja-se analisar no site.

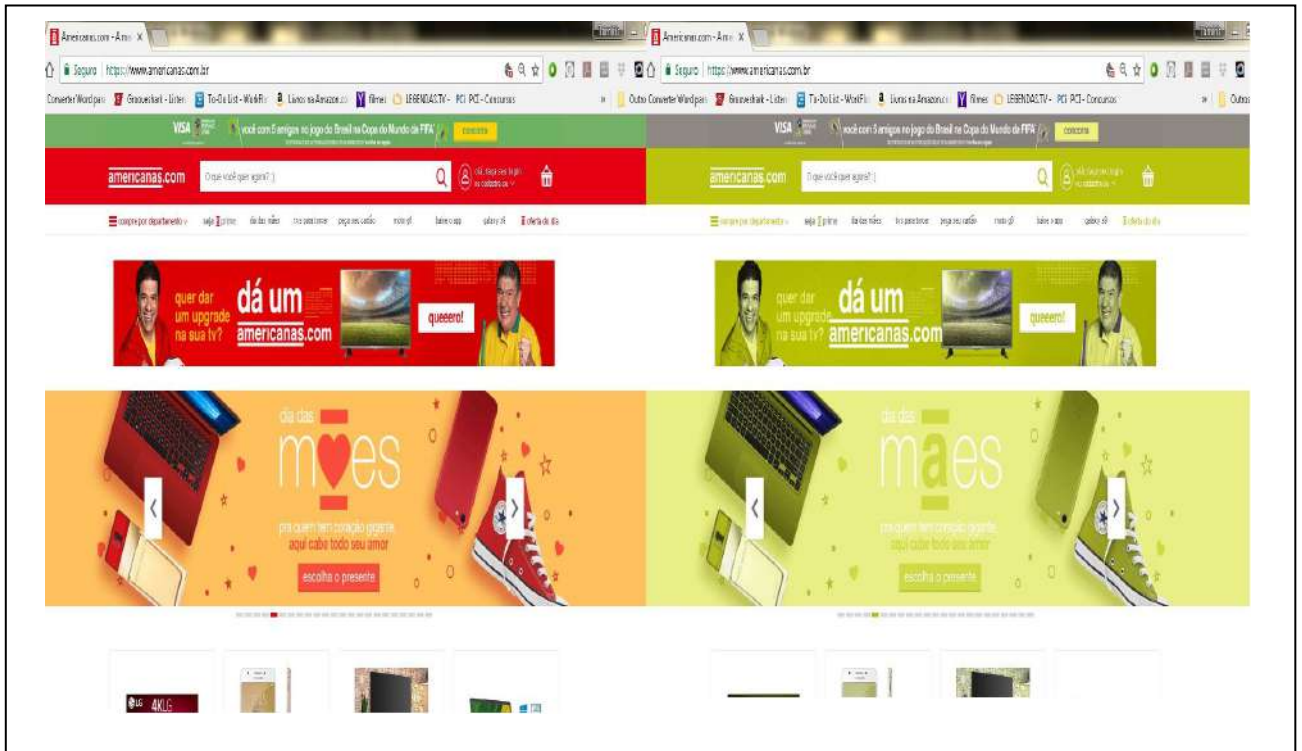
Figura 11: Menu do Simulador [NoCoffe Vision](#)



Fonte: [NoCoffe Vision](#)

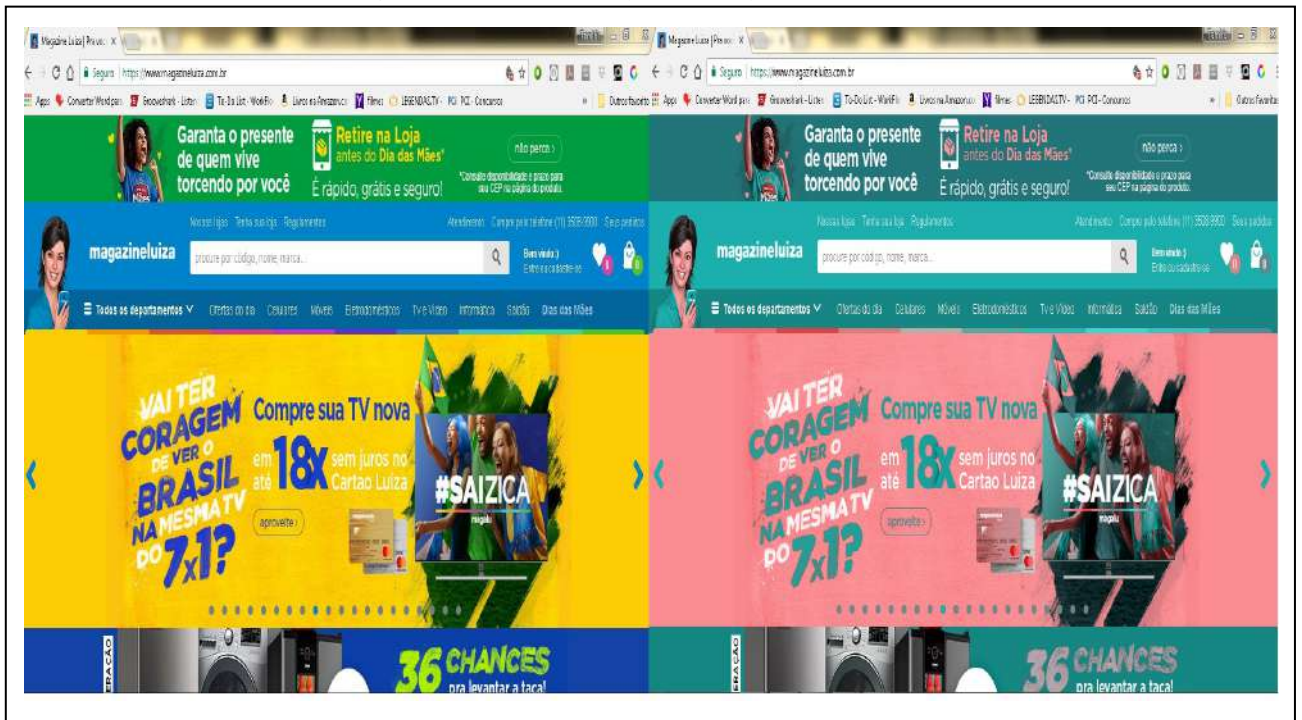
Para demonstração desta ferramenta utilizaremos os sites de comercio eletrônico: [Americanas.com](#) , [Saraiva.com.br](#) e [MagazineLuiza.com.br](#), onde em cada site será aplicado um filtro de visualização equivalente a um dos três tipos de daltonismo mais comuns citados anteriormente, com a intenção de demonstrar o comportamento destas interfaces para os usuários (Figuras 12, 13 e 14).

Figura 12: Filtro da visão por Protanopia



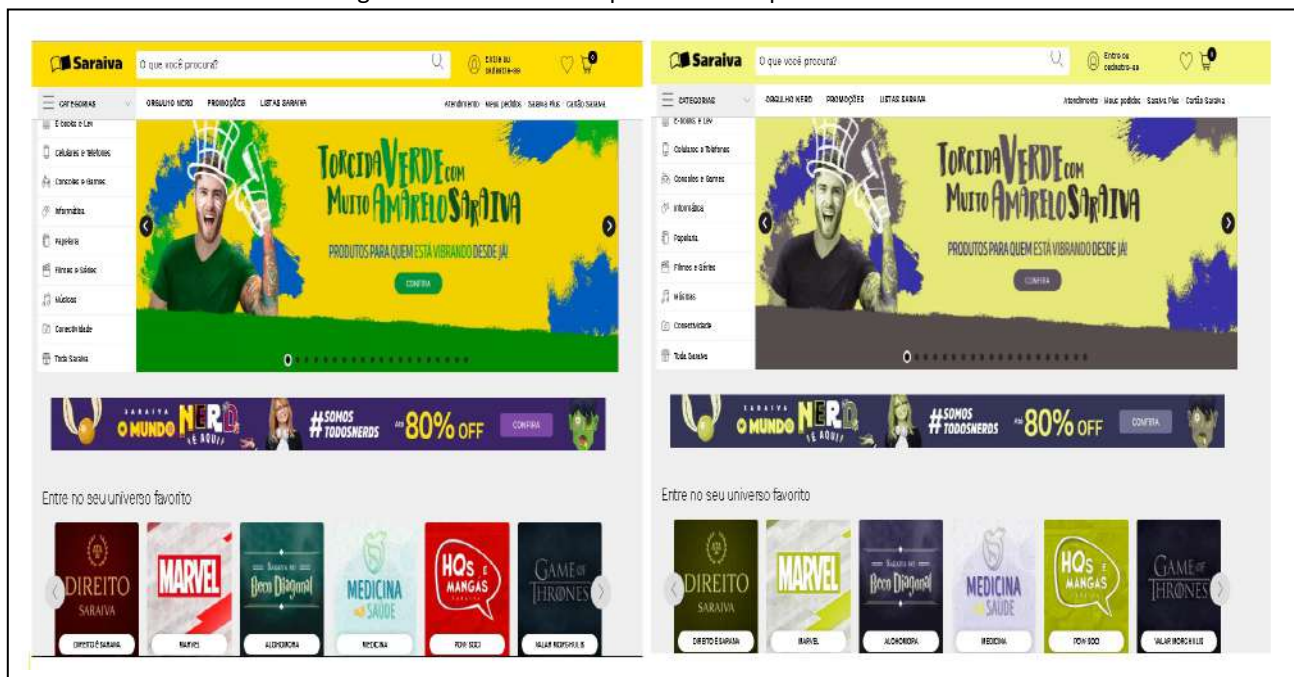
Fonte: Site Americanas.com

Figura 13: Filtro da visão por Tritanopia



Fonte: Site Magazine Luiza

Figura 14: Filtro da visão por Deuteranopia



Fonte: Site Saraiva.com.br

Aplicando-se os filtros que simulam a visão por usuários com Protanopia, Tritanopia e Deuteranopia que são respectivamente deficiência parcial dos cones vermelhos, azuis e verdes pode se observar que as propagandas perdem um pouco de sentido sem a devida identificação da cor, nos sites das lojas americanas na propaganda voltada ao dia das mães, usa-se de itens em tons vermelhos e derivados que relacionam a compra dos produtos ao amor e ou carinho materno.

Já na propaganda da Magazine Luiza e Saraiva a intenção está voltada a copa do mundo que ocorrerá em meados do ano 2018, procura impulsionar o torcedor brasileiro à compra de produtos relacionados a copa, usando das cores azul verde e amarelo para relacionar o usuário ao patriotismo brasileiro.

Usando desta ferramenta, podemos identificar algumas das dificuldades apresentadas em e-Commerce, que poderiam ser minimizadas se o desenvolvedor usa-se destas para criar suas interfaces gráficas, buscando a criação de projetos que ao se depararem com usuários cegos a cor, não tenha total perda de sentido visual.

Porem embora a ferramenta NoCoffe Vision auxilie bastante o desenvolvedor a entender como sua interface será vista na web, ela não ajuda quando o projeto ainda não está online, pois esta é uma extensão que só pode ser aplicada através de um navegador, não permitindo ser usado na tela do computador em outros testes fora do browser.

6.2 Simulador de Tela

A ferramenta Color Oracle pode ser usada para auxílio do projeto inicial, este é um simulador de tela capaz de simular a visão dos daltônicos em tudo que é exibido na tela do computador, ele permite que o desenvolvedor avalie seu projeto ainda na criação do protótipo. Depois de instalado este simulador, pode ser ativado a qualquer momento, o botão de ativação deste, fica na barra de tarefas do computador e quando iniciado a simulação, sua tela é congelada e todas as cores são convertidas imediatamente de acordo com a opção escolhida no menu.

O uso desta ferramenta proporciona agilidade na hora de testar seu projeto, se esta com duvidas e neste momento não têm a intenção de rodar seu projeto no browser, pode usar esta para ter uma resposta rápida e pratica da interface atual da aplicação, porem o Color Oracle é estático, não permitindo a interação com a tela que foi supostamente congelada, ele permite apenas a visualização da tela em que ele foi ativado no seu computador e qualquer movimentação da mesma, fará com que a simulação seja desativada.

Usar destas técnicas para desenvolvimento assistivo proporciona um maior índice de assertividade no impacto que o e-commerce irá proporcionar ao usuário, mas limita este ao ato de desenvolver. Considerando que estas técnicas ajudam, mas não são completamente eficazes na hora da interação entre o usuário e a interface, entende-se que estas ferramentas proporcionam simulações, que podem não ser totalmente fieis a visão de um daltônico. Pensando nisto, será apresentada uma ferramenta que auxiliam não só os desenvolvedores, mas também os usuários.

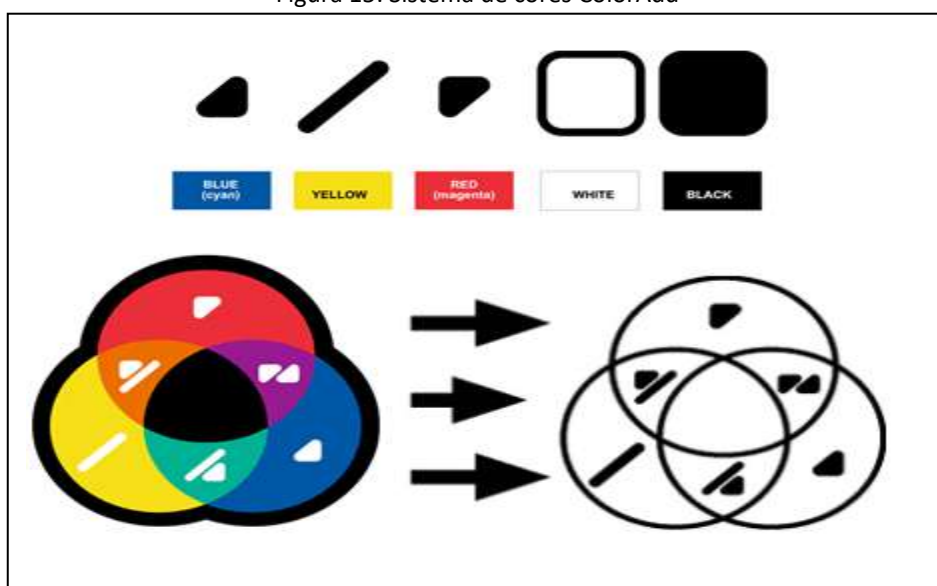
7 SISTEMA DE IDENTIFICAÇÃO DE CORES

Tendo em vista a dificuldade da cegueira parcial ou total de visão ou de cores na sociedade, estudiosos buscam minimizar estas através de códigos que referenciam as cores em forma de símbolos, projetos como Feelipa Color Code criado pela portuguesa Felipa Pires onde se atribui formas geométricas a cores e o ColorAdd criado também por um português chamado Miguel Neiva baseado em símbolos gráficos que auxiliam os daltônicos a distinguir as cores, são a prova disto.

7.1 Sistema Coloradd

O sistema de cores ColorAdd é um sistema com base em três símbolos gráficos básicos, usados para identificar cada uma das cores primárias e as cores branco e preto que surgem para distinguir os tons claros e escuros. Identificar cada um destes símbolos ou cada combinação possível entre eles possibilita a visualização das cores aos daltônicos conforme esquema na figura 15.

Figura 15: Sistema de cores ColorAdd



Fonte: Site Coloradd.net

Este projeto já foi aplicado em Portugal em uma marca de lápis de cor, na identificação de setores e seguimentos de hospitais, mapas de metros, etiquetas de roupas e até mesmo em jogos, este recebeu o premio das nações unidas por promover a igualdade, entre outros reconhecimentos. Sendo a internet propulsora da comunicação atual, ela não poderia ficar fora desta e foi pensando em como aplicar este código na web que a empresa Noesis se juntou ao colorAdd em busca da aplicação desta tecnologia aos usuários de web sites.

7.2 Ferramenta Coloradd Web Picker

Desta junção surgiu o colorAdd Web Picker, um sistema em java criado para identificar as cores de uma página, que funciona basicamente como um coletor destas, possibilitando sua identificação em tempo real nos web sites. Os que possuem esta tecnologia terão em seu menu principal o símbolos do coloradd como identificador da ferramenta, geralmente localizado na parte superior do menu. No momento em que esta ferramenta é ativada, surge na lateral da página a caixa de menu, onde é possível interagir com a ferramenta, conforme demonstra a figura 16.

Figura 16: Menu da aplicação Coloradd Web Picker

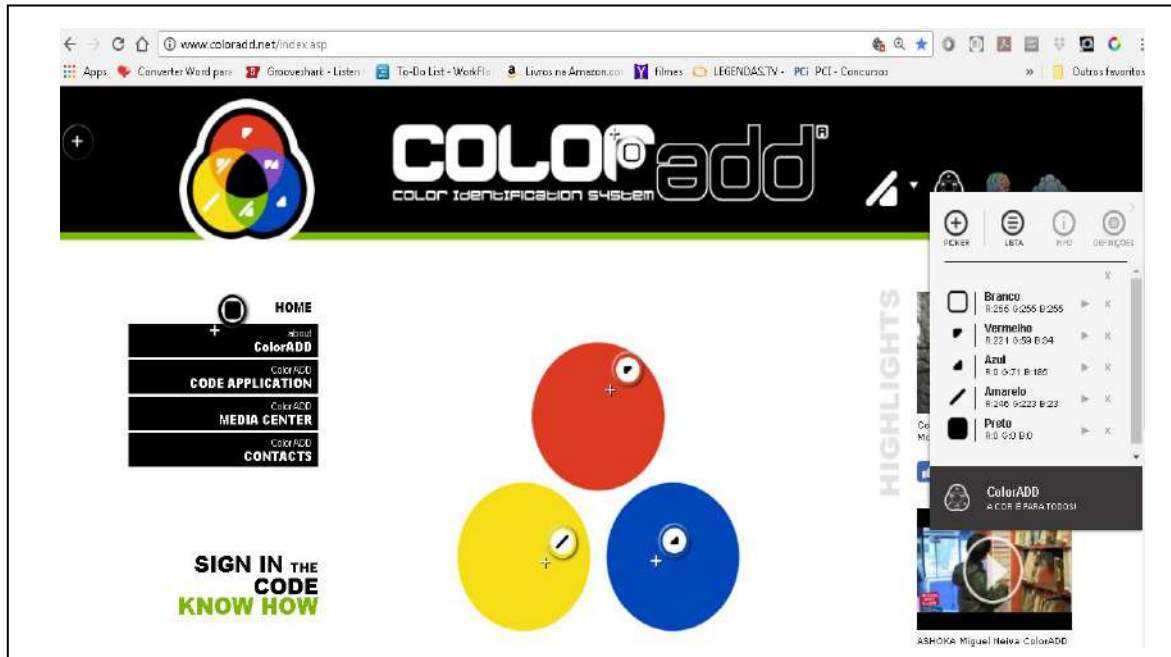


Fonte: Site Coloradd.net

Pode-se usar esta aplicação através do modo simples ou avançado, onde ambos disponibilizam uma mira que ira captar a cor desejada. Na mira de modo avançado, pode-se escolher a forma de linguagem a ser aplicada no sistema e o usuário pode ativar a identificação sonora da cor, onde uma gravação descreverá qual cor foi selecionada ou desativar esta, aplicando-se apenas um símbolo do código coloradd equivalente a cor, no local clicado.

Todas as cores selecionadas serão disponibilizadas na área de trabalho do menu desta ferramenta, no campo lista e poderão ser visualizadas ou ouvidas a qualquer momento durante a navegação atual, conforme demonstrado na figura 17 em um teste realizado no próprio site coloradd.net.

Figura 17: Teste da Aplicação Web Coloradd



Fonte: Site Coloradd.net

Este é um sistema que ajuda a proporcionar igualdade na navegação é mais uma forma de trazer o desenvolvimento assistivo à web e pode ser implementado através de uma licença de utilização onde o valor é ajustado de acordo com o volume de negocio de cada parceiro. Com tudo o sistema ainda possui algumas falhas de utilização, se o usuário sair da página na qual iniciou a verificação das cores, a lista é a pagada e o sistema é desativado imediatamente sendo necessitando uma nova inicialização na página atual.

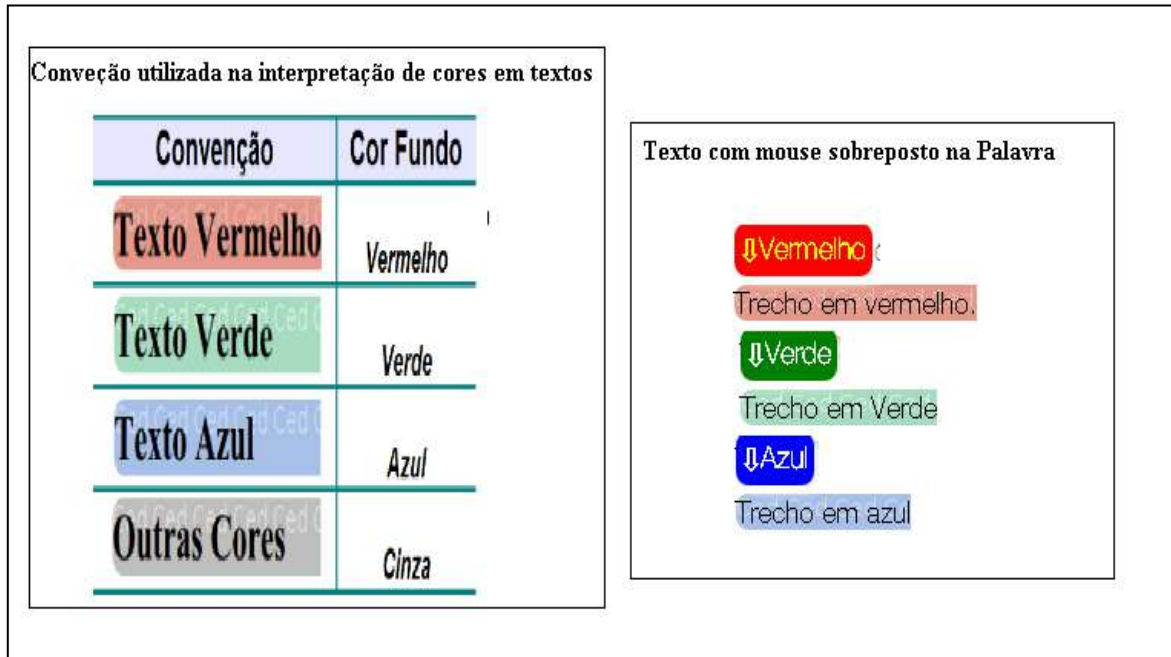
Este é um sistema pouco utilizado no momento e não foi encontrada nenhuma loja e-commerce nacional ou internacional que use esta tecnologia para proporcionar interação e inclusão de daltônicos à loja.

7.3. Conversor de Estilo

Foi proposto por Amauri Duarte (2013) no congresso ConSerpro, a criação do CED - Conversor de Estilos para Daltônicos, esta ferramenta teria a função de receber todo o estilo *color CSS* de um site, e converter este a um estilo adaptado.

Nesta todos os textos coloridos existentes em uma página web, deveriam possuir uma legenda que especificasse qual cor estava sendo usada no texto ao ativar a folha de estilo CSS convertida, seria aplicado um fundo em cada palavra escrita em cores diferentes de preto e cinza e quando o usuário sobrepusesse a palavra com o mouse, o nome da cor seria apresentado.

Figura 18: Técnica para identificação de cores em textos desenvolvida por Amauri



Fonte: Site www.daltonicos.com.br

Amauri relata em seu trabalho, a importância desta técnica de conversão, deixando claro que o estilo pode ser aplicado sem o uso da CED, mas que esta ferramenta trará maior alcance da técnica sugerida. A agência de inovação da Sempro iniciou a elaboração de um portfólio de projetos inovadores e se propôs a desenvolver esta ferramenta, porém ela ainda não está disponível.

Usa-se muito das cores no texto, para identificação de links, demonstrar se o mesmo já foi ou não clicado, enfatizar palavras importantes e até mesmo enfeitar propagandas, ter um estilo que proporciona a identificação da cor destes textos, será eficaz a os usuários que necessitam.

No quadro a baixo, pode ser observada de forma resumida a avaliação das técnicas e ferramentas abordadas neste trabalho:

Tabela 1: Comparação de técnicas/simuladores

Ferramentas e Técnicas	Ponto negativo	Ponto Positivo
Folha de estilo Css alternativa com opção de contraste	Não se aplica nas imagens como propagandas, logotipos, gráficos, entre outras, que compreendem o corpo do site minimizando o alcance da interface. Pode reduzir a criatividade do designer.	Fácil de ser aplicado em qualquer site.
NoCoffe Vision	Funciona apenas no browser	Alcança todas as cores e imagens do site, permite a interação durante toda a simulação e é capaz de simular não só a visão de daltônicos

Color Oracle	Simulação estática, não permite interação com a tela quando ativado.	Permite simular qualquer tela do computador, auxilia no desenvolvimento ainda em fase de protótipo.
Coloradd Web Picker	A lista de cores é encerrada a cada mudança de página. Direitos da ferramenta são reservados e precisa ser comprado.	Ajuda o usuário com a identificação da cor em tempo real, criando maior interação com o site.
CED - Conversor de Estilo para Daltonico	Ferramenta ainda não desenvolvida.	Permite a conversão de todas as folhas de estilo CSS de forma ágil, aplicando legenda a todo o texto colorido do site.

Estes são apenas alguns dos exemplos de simuladores e técnicas que podem ser usadas para permitir a inclusão e a boa navegação dos daltônicos na web, usar das cores é quase sempre a primeira opção de designers quando querem evidenciar uma informação ou direcionar seus usuários a um local específico de forma rápida, mas esta tática nem sempre é eficaz. É aconselhável usar sempre legendas, ícones e palavras, ter o devido cuidado com o contraste entre as cores, informações em alto contraste permitem maior reconhecimento aos olhos de quem confunde as cores.

8. CONSIDERAÇÕES FINAIS

Este trabalho explorou o crescimento abundante do e-commerce no mercado, delineando suas fases. Identificou os potenciais compradores de acordo com suas gerações que ajudam a definir características específicas dos usuários. Demonstrou o impacto econômico sobre esta plataforma de venda que cresce a cada dia e poderá ser indispensável futuramente.

Com a identificação dos usuários, considerando a grande quantidade de pessoas com cegueira à cor, apresentou uma análise de algumas técnicas de desenvolvimento e sistemas existentes que auxiliam desenvolvedores front-end e ou designers web a desenvolverem lojas e-commerce acessíveis tendo em vista a importância das cores nos projetos e a porcentagem considerável de usuários daltônicos no mundo. Demonstrou soluções de TAs que podem ser aplicadas no desenvolvimento destas lojas

Realizou o comparativo das TAs que possibilitam a inclusão dos grupo dos três tipos de daltônicos mais comuns: protanomalia, deuteranomalia e tritanomia; na web, demonstrando o ponto positivo e o negativo de cada TA apresentada.

Buscou-se aprimorar a visão sobre a aplicação das cores em um projeto, não sendo considerada apenas a opção de contraste entre as cores nos tons de branco, preto e cinza, que por sua vez podem minimizar a criatividade do projeto. Procurou incentivar o desenvolvimento inclusivo e facilitar o trabalho dos desenvolvedores acerca de técnicas vigentes para proporcionar acessibilidade e inclusão, analisando e apresentando ferramentas existentes.

Com este foi possível observar que a prática de desenvolvimento assistivo não é tão usada como deveria, buscou demonstrar a necessidade deste desenvolvimento nas interfaces gráficas de lojas e-commerce, visando à interação e inclusão do público consumidor daltônico com esta plataforma de venda.

9. REFÊRENCIAS BIBLIOGRÁFICAS:

ABRANTES José Carlos, Breves, **contributos para uma ecologia da imagem**, BOCC, 1999. 12p. Disponível em: < www.bocc.ubi.pt/pag/abrant-es-jc-ecologia-imagem.pdf> Acessado em: 10 de Março 2018.

AMBROSE Gavin; HARRIS Paul, **Design básico: tipografia**. Porto Alegre: Bookman, 2011, 180 p.

ANDRADE, Gilmar Vitor da Silva; PINTO, Joseh Augusto Dantas Salgado; SILVA, Bruno Santana da; "Simuladores e Tecnologias Assistivas para Daltonismo", p. 1966-1977 . In: . São Paulo: Blucher, 2017. ISSN 2318-6968, DOI 10.5151/16ergodesign-0207 Disponível em: < <https://www.proceedings.blucher.com.br/article-details/simuladores-e-tecnologias-assistivas-para-daltonismo-25864>> Acessado em: 13 de Setembro de 2017.

ALBERTIN, A. L. **Comércio Eletrônico: Modelo, Aspectos e Contribuições de sua aplicação**. São Paulo: Atlas, 2004. 318 p.

Brasil. **Subsecretaria Nacional de Promoção dos Direitos da Pessoa com Deficiência**. Comitê de Ajudas Técnicas. Tecnologia Assistiva . – Brasília: CORDE, 2009. 138 p.

CENSO demográfico, Rio de Janeiro, **IBGE**, 2010. 215p. Disponível em: <https://biblioteca.ibge.gov.br/visualizacao/periodicos/94/cd_2010_religiao_deficiencia.pdf> Acessado em: 14 de Abril 2018.

CERETTA B. Simone; FROEMMING M. Lurdes, **Geração Z: Compreendendo os Hábitos de Consumo da Geração Emergente**, Rio Grande do Sul, 2011, 24p. Disponível em: <<https://repositorio.unp.br/index.php/raunp/article/view/70>> Acessado em 24 de Março 2018.

Do R7, **Estudo revela que brasileiro passa mais de nove horas por dia na internet**, Portal R7 de Notícias, 2015. Disponível em: <<https://noticias.r7.com/tecnologia-e-ciencia/estudo-revela-que-brasileiro-passa-mais-de-nove-horas-por-dia-na-internet-23012015>> Acessado em: 05 de novembro 2017.

DUARTE, Amauri, **Acessibilidade para Daltônicos na Web**, Disponível em: < <http://www.daltonicos.com.br/daltonico/index.html>> Acessado em: 14 de maio 2018.

E-commerce News, **Conheça um pouco da história do e-commerce [Infográfico]**. Redação E-Commerce News, 2011. Disponível em: <<https://ecommercenews.com.br/noticias/balancos/conheca-um-pouco-da-historia-do-e-commerce-infografico/>>. Acesso em: 29 de outubro de 2017.

EBIT, **Webshoppers 35ª Edição 2016**. Disponível em: <<http://www.ebit.com.br/webshoppers>. Acessado em: 05 de outubro de 2017>. Acessado em: 03 de Março 2018.

EBIT, **Webshoppers 36ª Edição 2017**. Disponível em: <<http://www.ebit.com.br/webshoppers>. Acessado em: 03 de março de 2018>. Acessado em: 03 de Março 2018.

EBIT, **Premio Ebit Melhores do E-commerce**, Disponível em: <<http://www.ebit.com.br/premio-ebit/2018>> Acessado em: 03 de Março 2018.

FILIPINI Dailton, **Abc do e-commerce**, Le-books 2011. Disponível em: <<https://www.passeidireto.com/arquivo/23269526/abc-do-e-commerce>>. Acesso em: 10 de Março de 2018.

GUIMARÃES B. J. Italo, SOUZA F. R. Marckson, **Acessibilidade em Websites de comércio Eletrônico: Avaliação através da intervenção com usuários cegos na Paraíba**. Pesq. Bras. em Ci. da Inf. e Bib. 2015, João Pessoa, v. 10, n. 1, 197p. Disponível em: <www.periodicos.ufpb.br/index.php/pbcib/article/viewFile/24559/13438> acessado em: 12 de Dezembro de 2017.

GUIMARÃS Luciano, **A cor como informação**, São Paulo: Annablume, 2000, 160 p.

LINS, Dulciane Torres. **Geração Y: o nascimento de uma nova versão de líderes**. Sidnei Oliveira. São Paulo: Integrare Editora, 2010. Revista Científica Hermes - FIPEN, [S.l.], v. 4, jan. 2011. ISSN 2175-0556. Disponível em: <<http://www.fipen.edu.br/hermes1/index.php/hermes1/article/view/43/112>>. Acesso em: 24 mar. 2018.

LORDELO Antonio Luiz Andrade, **Usabilidade de Interfaces Web**, E-papers Serviços Editoriais LTDA, 2007, 124p.

MEGER Botelho Gabriela, Interface Gráfica para Aplicativo Computacional Florianópolis **Dissertação** 2004, 110 p. Disponível em: <<https://repositorio.ufsc.br/handle/123456789/87584>>, acessado em: 12 de Outubro de 2017.

MELO Débora Gusmão, Galon, José Eduardo Vitorino e Fontanella, Bruno José Barcellos. Os "daltônicos" e suas dificuldades: condição negligenciada no Brasil?. Physis: Revista de Saúde Coletiva [online]. 2014, v.24, n. 4 Acessado 15 Abril 2018, pp. 1229-1253. Disponível em: <<https://doi.org/10.1590/S0103-73312014000400011>>

NAKAMURA, R.R. **E-Commerce na Internet: Fácil de Entender**. São Paulo: Érica, 2001. 272 p.

NISSAN Mauro, Qual a diferença entre B2B e B2C. **E-commerce News**, 2014. Disponível em: <<https://ecommercenews.com.br/artigos/cases/qual-e-a-diferenca-entre-b2b-e-b2c/>> c

PEDRASSOLLI Leandro; NERIS Vânia, **O Uso de Cores em Aplicações Web: um Estudo dos Projetos Desenvolvidos no Curso Lato Sensu de Desenvolvimento de Software para a Web**, UFSCar, 2014 Disponível em:

<<http://revistatis.dc.ufscar.br/index.php/revista/article/view/91>> acessado em: 11 de março 2018

ROUSEEF Dilma, **Lei federal N° 13.146**, Brasília 2015, 34 p. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/113146.htm> acessado em: 14 de Abril 2018.

SÁ De Sylvia, Geração Z: quem são os consumidores do futuro, Materia da Exame em outubro de 2010 Disponível em: <<https://exame.abril.com.br/marketing/geracao-z-quem-sao-consumidores-futuro-596163/>> Acessado em 24 de março de 2018.

SADRONE Laura C., De lobato a **Bojunga: as reinações renovadas 2ed**, Rio de Janeiro: Nova Fronteira 2011, 192 p.

SCANDIUZZI Fernando, OLIVEIRA Márcio Mattos Borges, ARAÚJO Geraldo José Ferraresi de, **A Logística no Comercio Eletrônico B2C um Estudo Nacional Multi Casos**, Caderno de Administração - v.19, n.1, 2011. Disponível em: <<http://periodicos.uem.br/ojs/index.php/CadAdm/article/view/13050>> Acessado em: 10 de março 2018.

SILVA, Quelita Araújo Diniz; SILVEIRA, Maria Augusta; Netto Nunes; OLIVEIRA, Adicinéia Aparecida, **Respeito à Cidadania: Promovendo Acessibilidade Web na Universidade, Federal De Sergipe (UFS)**; Revista GEINTEC ISSN 2237-0722, São Cristovão - SE, 2012 Vol.2, N.1, p. 82-99. Disponível em: <<http://www.revistageintec.net/index.php/revista/article/view/28/65>> Acessado em: 10 de março 2018

TOREZANI Nathália, N. **O crescimento do e-commerce no Brasil**. Revista iMasters, 2008. Disponível em: <<http://imasters.com.br/artigo/9649/e-commerce/o-crescimento-do-e-commerce-no-brasil/>>. Acesso em: 03 de Março de 2018.

TURCI Fabio, **Gerações apresentam diferentes perspectivas e metas profissionais**, São Paulo, G1, 2010 Disponível em: <g1.globo.com/jornal-da-globo/noticia/2010/11/geracoes-apresentam-diferentes-perspectivas-e-metas-profissionais.html> Acessado em: 24 de Março 2018.

PROTOCOLO DE APLICAÇÃO

DDP: *Distributed Data Protocol*

Vinicius Ataide de Albuquerque
Sheyla Natália de Medeiros

RESUMO

O protocolo DDP (*Distributed Data Protocol*) de sistemas web é um protocolo implantado na plataforma MeteorJS para que aplicações web estejam sempre atualizadas perante o usuário evitando inconsistência na informação. Diferente do HTTP (HyperText Transfer Protocol), o protocolo DDP é baseado em conceitos de aplicações modernas utilizadas atualmente, através de celulares com aplicativos *mobile* ou na web através do *browser*. Aplicações como o *Facebook* ou *Messenger* funcionam semelhante tanto no celular quanto no desktop graças à arquitetura de comunicação entre o cliente-servidor, contudo essa arquitetura foi desenvolvida para a infraestrutura do aplicativo, não sendo genérica o bastante para se encaixar a qualquer regra de negócio. O DDP é genérico o bastante a nível de trabalhar com mensagens e faz com que o desenvolvedor não precise criar uma infraestrutura para que se entendam, pois diferente do HTTP que só dispõe de dois métodos (*GET* e *POST*, excluindo os verbos *REST PUT*, *PATCH*, *DELETE* e *UPDATE*, aos quais foram introduzidos em outros *RFCs*), o DDP na própria concepção foi pensado no cliente com várias funcionalidades (cache, orquestração com o servidor, isolamento de informação sigilosa do usuário), isso é importante para o desenvolvedor pois o isola de complexidades que sempre têm de pensar quando desenvolve uma aplicação web rica em interação com o usuário. Este relatório tem como objetivo abordar as diferenças e vantagens do uso do DDP no contexto de WebApps em relação ao HTTP com REST (Representational State Transfer), demonstrando a dinâmica na conversação do cliente com o servidor e de como à formulação do projeto conceitual da aplicação é simplificada para o desenvolvedor. É desenvolvido em anexo um sistema em código aberto para garantir o controle de entrada e saída de visitantes na Procuradoria Federal da República do Estado da Paraíba e depois relatórios são feitos como demonstração da eficácia no uso do DDP em comparação ao HTTP. Vale salientar que o HTTP não deve ser considerado concorrente, e sim complemento do DDP.

Palavras-chave: Protocolo, WebApps, Desenvolvimento, HTTP, DDP, Aplicações, Cliente, Servidor.

1 INTRODUÇÃO

Protocolos de aplicação, de acordo com o modelo OSI (Open Systems Interconnection), são a barreira mais próxima do desenvolvedor para a arquitetura de rede de computadores. Provê a semântica necessária para documentar todas as funcionalidades externas do sistema, tornando-se essencial como conhecimento de construção de aplicações distribuídas.

De acordo com Sashko Stubailo e sua especificação do protocolo DDP, o mesmo funciona em cima do protocolo HTTP, assim como o protocolo *WebSocket* (outro

protocolo de aplicação), demonstrando que os mesmos podem ser empilhados a fim de manterem suas funcionalidades, adicionando-as subsequentemente ao protocolo abaixo, sendo assim uma importante escolha na pilha de tecnologias que o desenvolvedor deseja usar.

Este documento aborda uma analogia com sistemas operacionais antigos e modernos, explicando o motivo da evolução dos sistemas operacionais analogamente à evolução tecnológica da época, prosseguindo com uma introdução ao protocolo HTTP e um detalhamento acerca do protocolo DDP, encerrando com uma comparação performática entre os dois protocolos.

JUSTIFICATIVA

A escolha do DDP como protocolo é motivada por 3 fundamentos de acordo com Matt Debergalis (2015, criador do protocolo). Performance, sendo apenas uma requisição enviada para o servidor como um pedido de troca de protocolo para o *WebSocket* (atualmente o padrão web) como é demonstrado na figura 1 abaixo, causando a diminuição do tráfego geral pois não há handshake entre cada recurso como é no protocolo TCP (*Transmission Control Protocol*).

Arquitetura, pois uma aplicação movida a HTTP/REST deve ser construída em torno de novos protocolos, que diferem dos protocolos utilizados na aplicação nativa de um SO (Sistema Operacional). Com o DDP a arquitetura de uma aplicação fica análoga à uma nativa.

Semântica, Um POST na rota “/arquivos” é diferente semanticamente de um método “adicionar-arquivo”, tradicionalmente utilizado no protocolo RMI (Remote Method Invocation).

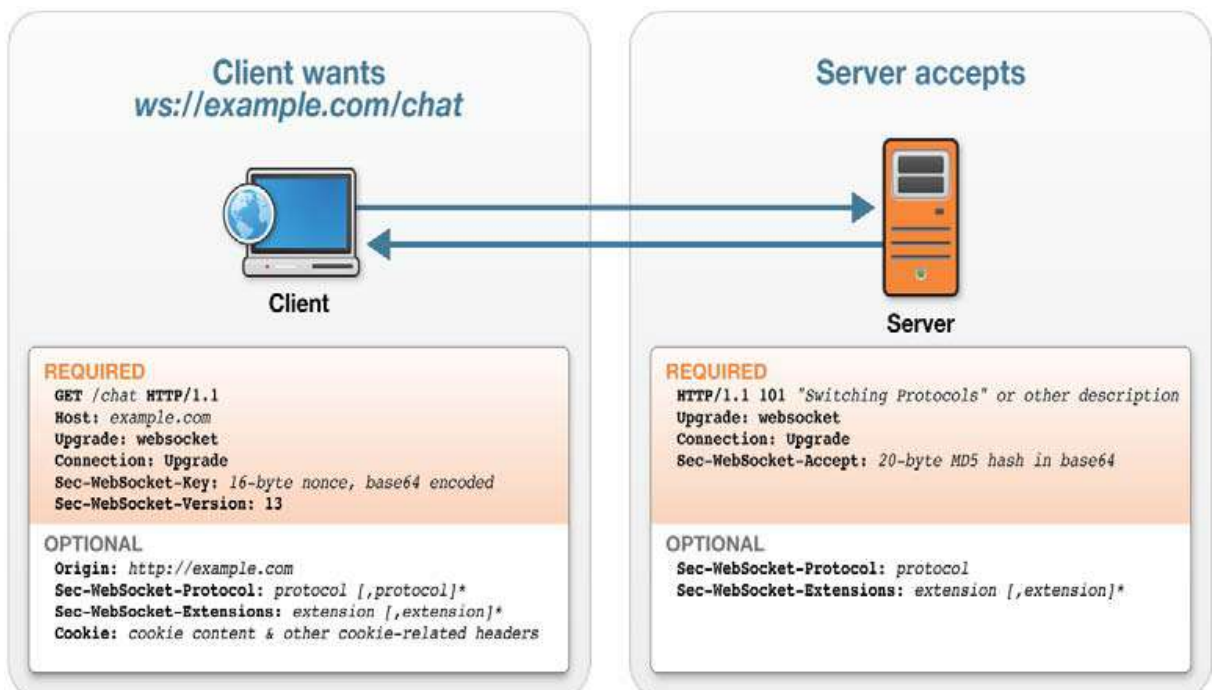


Figure 1: Pedido de inicialização de uma conversa *WebSocket*.

Fonte: <http://apress.jensimmons.com/v5/pro-html5-programming/ch7.html>, acesso em 10/13/2016

O DDP não é o único protocolo que tenta resolver as dificuldades do REST, várias outras soluções existem no mercado estão disponíveis, o GraphQL do Facebook e o Falcor da Netflix, sendo essas ferramentas mais inteligentes de integração com

servidores, pois delegam ao cliente o formato da informação que deseja, o GraphQL é um *parser* com uma metalinguagem embutida como forma de enriquecimento na comunicação, através de uma DSL (Domain-Specific Language) muito parecida com JSON.

2 OBJETIVOS

2.1 OBJETIVO GERAL

Apresentar o protocolo DDP (Distributed Data Protocol), utilizando tecnologias web como fator prático, ressaltando a eficiência tanto no desenvolvimento quanto na implantação do sistema.

2.2 OBJETIVOS ESPECÍFICOS

1. Embasar o DDP historicamente, teoricamente e no uso prático através de aplicações.
2. Utilizar ferramenta Meteor para demonstrar o DDP contra o JSF com HTTP/REST.
3. Comparar o DDP com o HTTP/REST; demonstrar com gráficos comparativos o tamanho das mensagens trocadas por uma aplicação antiga e moderna utilizando os dois protocolos; demonstrar o uso do TCP com esses dois modelos.

3 SISTEMAS OPERACIONAIS, APLICAÇÕES E PROTOCOLOS

De acordo com Tanenbaum, antes dos anos 80 na terceira geração e anteriores de SOs (Sistemas Operacionais) os computadores de mesa eram terminais burros, não tinham poder computacional, só redirecionavam a entrada do usuário a um servidor que fazia a toda a operação, assim chamados sistemas monolíticos¹, pois só havia uma implementação do trabalho para todos os terminais, aquela do mainframe.

Esse tipo de aplicação difere arquiteturalmente das aplicações nativas atuais, pois antigamente os conceitos de engenharia distribuída não existiam tanto quanto protocolos de conversação definidos mundialmente, cada mainframe trabalhava isoladamente em uma empresa, Tanenbaum A. S. descreve a evolução dos conceitos de forma cíclica na seguinte frase:

A ciência da computação, assim como muitas outras áreas, é bastante orientada pela tecnologia. Os antigos romanos não tinham carros não porque gostassem de andar, mas porque não sabiam construir carros. Computadores pessoais existem não porque milhões de pessoas desejassem insaciavelmente possuí-los, mas porque tornou-se possível fabricá-los à baixo custo. (TANENBAUM 2009 A. S. pg. 23)

De acordo com Matt Debergalis (2013), Aplicações que seguem o modelo HTTP (sem REST) cliente-servidor seguem a mesma proposta dos mainframes, detendo toda a computação no lado do servidor deixando o cliente apenas a receber um *template*

¹ De acordo com o dicionário Michaelis: Obra ou monumento elaborado ou esculpido em um só bloco de pedra.

do resultado da operação computada, neste sentido, o browser é o terminal e o servidor o mainframe.

De acordo com a plataforma InfoQ Matt Debergalis é o CEO do MeteorJS, criador do protocolo DDP. Ele é um cientista da computação que contribuiu inicialmente para projetos *Open Source* como o projeto NeXT para o NetBSD e trabalhou na construção de especificações para o DAFS (*Direct Access File System*) e o NFSv4 (*Network File System Version 4*) sendo um hacker de *Kernel*, conseguiu recolher mais de 350 milhões de dólares em contribuição para a plataforma ActBlue, a maior plataforma de angariação de fundos política no planeta.

Matt descreve sua plataforma analogamente a sistemas operacionais modernos como o Unix ou *Windows* que utilizam conceitos dinâmicos de comunicação através de processos (denominadas aplicações). Essas aplicações se comunicam por sockets, sendo pensados como sendo para uso em computadores pessoais que não precisam de uma gerência distribuída completa. Toda a gerência é feita utilizando referências de processos, em outras palavras, nenhum processo detém o estado associado à informação, apenas uma referência na memória de um dado no sistema de arquivos ou no banco de dados.

Ele utiliza da analogia com Tanenbaum em Sistemas Operacionais Modernos explicando que quando o dado se disponibiliza para o processo em memória, a gerência dele é transparente para o processo, sendo em aplicações nativas o usuário alterando valores em um ponto da aplicação, é esperado que seja visto em todas as telas que referenciam aquele valor a mudança instantaneamente, por exemplo: Se uma *playlist* é adicionada no Windows Media Player, automaticamente você verá ela adicionada na sua lista de *playlists*, sem você precisar pedir o programa para recheicar as *playlists*.

Essa arquitetura é natural em aplicações nativas e se estende facilmente a outras aplicações no mesmo sistema através de memória compartilhada ou através de sockets. Através de eventos que uma aplicação envia para a outra, o Windows e qualquer outro sistema baseado em UNIX detém de várias API's com eventos para se comunicar com o sistema, esse modelo é denominado assíncrono bidirecional.

De acordo com Rich Hickey (2015) em "*Language of the System*" aplicações são como pequenos sistemas operacionais que se comunicam entre si através de bibliotecas como aplicações e chamadas de método como protocolos. Se uma aplicação não tiver essa concepção, tende a manter as tecnologias abordadas não adaptadas ao contexto do sistema. Com essa definição a aplicação se comporta como monolítica, no caso de aplicações que utilizam o HTTP, esse modelo de Rich Hickey se encaixa como monolítica.

Resolvendo as necessidades pode-se perceber a preocupação dos cientistas da computação em resolver essa inconsistência no desenvolvimento de sistemas nativos locais e em desenvolver aplicações web distribuídas, nesse sentido alguns protocolos vieram à tona na tentativa de resolver os problemas operacionais tão qual os conceituais que as grandes empresas com grandes aplicações web sofriam.

Dessa gama de tecnologias vale salientar duas tecnologias de integração cliente-servidor que se popularizaram rapidamente: O GraphQL do Facebook e o Falcor da Netflix. No caso do Falcor a forma de comunicação é baseada em JSON (*Javascript Object Notation*), já no caso do GraphQL ele utiliza uma linguagem de programação declarativa completa especializada nessa comunicação, com suporte à tipagem forte optativa, interfaces, fragmentos (que se assemelha à herança em OO (Orientado a Objeto)), e finalmente comandos (*mutations*) e queries como interação com o servidor.

Esses modelos trabalham o formato da mensagem e não a propagação dela, sendo adaptáveis perfeitamente ao uso do DDP, como será visto no capítulo 5 nas

mensagens trocadas, o foco está na propagação das mensagens, mas diante dessa complementação há uma semelhança na maneira em que ambos os protocolos separam as operações de busca e inserção/alteração/deleção no sistema, idéia original do protocolo CQRS.

O CQRS (*Command Query Responsibility Segregation*) como demonstrado na figura 2 é responsável somente pela segregação das queries dos comandos que as aplicações geram. Percebe-se uma semelhança nos conceitos adaptadas ao uso específico de cada protocolo, sendo eles DDP, CQRS, GraphQL.

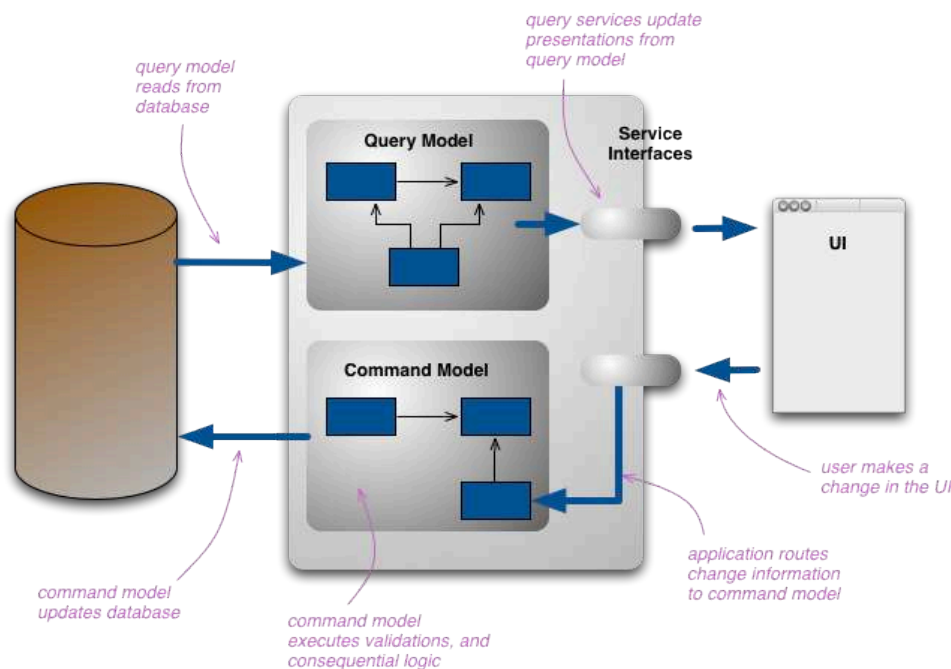


Figure 2: Protocolo CQRS.

Fonte: <http://martinfowler.com/bliki/images/cqrs/cqrs.png>. Acesso em 14/10/2016

Pode-se perceber a similaridade do conceito de queries do GraphQL com as características do CQRS, já que as Mutations se assemelham aos Commands, e as Queries às próprias Queries do GraphQL. A diferença nesses protocolos está apenas na nomenclatura dos elementos e especificidade das funções, pois o GraphQL especifica a estrutura da mensagem enquanto o CQRS especifica a divisão de responsabilidades entre Queries e alterações impactantes no sistema.

Veremos adiante onde o protocolo DDP se inspirou nos conceitos dos sistemas operacionais modernos e como desenvolver nele leva vantagens no custo do tempo de desenvolvimento no sistema.

4 HTTP (HyperText Transfer Protocol)

Aplicações criadas em Frameworks HTTP desenvolvem abstrações em cima do protocolo para facilitar o trabalho das regras de negócio para o desenvolvedor, pois o protocolo de acordo com o RFC 2616 foi criado para desenvolvimento de hipertextos, e não de aplicações, criando uma barreira maior para retirada de bugs, pois o código que é digitado entra em uma pilha de abstrações e camadas de ciclos de vida em cima do próprio HTTP. Veremos a seguir um framework que utiliza o HTTP como protocolo de transporte e implementa em cima dele várias abstrações e ciclos de vida próprios.

4.1 JSF (Java Server Faces)

O JSF é um *framework* Java que se popularizou pela forma de trabalhar as interações com o usuário, em vez de trabalhar diretamente o protocolo HTTP (com o doGet e o doPost no servlet) ele criou o conceito de *managed beans* (*controllers*) sobre *facelets* (*views*), o sistema funciona todo a base de eventos, como no caso do delphi sobre o pascal, resultando em uma interação mais fragmentada, pois os eventos não se limitam à navegação do site, e sim também a clique de botões, preenchimento de entradas, ou envios de formulários.

Mesmo com esse aumento no vocabulário de interações, o fundamento do mainframe ainda permanece, pois todos os eventos são enviados ao servidor e só ele consegue dar razão àquele valor recebido, mantendo ainda o conceito de browser como terminal burro, essa característica ainda é mais forte quando vemos como tudo funciona por trás, a aplicação é guardada no servidor pois nada existe no cliente (através da sessão javascript), todas as referências dos valores estão em atributos XML (*Extensible Markup Language*) do *facelet* (*view*) que são lidas no *managed bean* (*controller*).

Vê-se então uma grande quantidade de conteúdo trocado entre servidor-cliente, conteúdo esse que não é HTML, forma ainda não esperada pelo HTTP. Nesse cenário um novo protocolo surgiu pela Microsoft e se popularizou, apesar de implementação diretas em código complicada para os desenvolvedores em suas aplicações, O XHR (XML HTTP Request) serviu como solução pois era para essa troca de informações e não de telas HTML.

A partir dessa forma de trabalhar, o ciclo de vida de requisição HTTP não tem mais sentido na aplicação, sendo necessário um novo ciclo de vida, como demonstrado na figura 2. Neste ciclo de vida, a requisição e a resposta residem nas extremidades do fluxo, sendo elas para cada evento no cliente, enquanto todos os eventos de importância para aplicação são invocados quando necessário no processo.

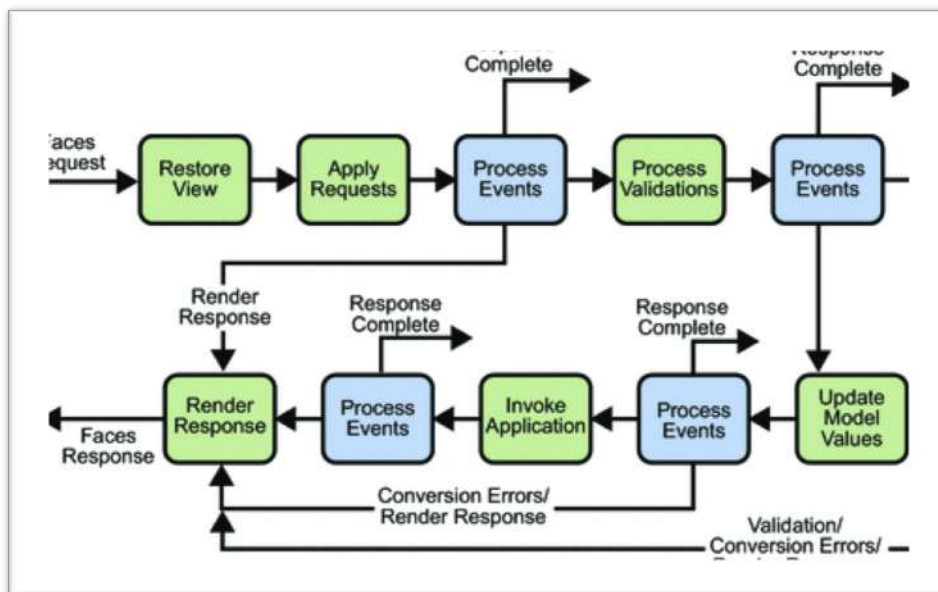


Figure 3: Ciclo de vida JSF.

Fonte: <http://www.devmedia.com.br/ciclo-de-vida-do-javascript-server-faces-jsf/27893> acesso em 04/09/2016

Diante dessa complexidade de conceitos, vários esforços se concentram para simplificar a maneira de pensar como sistemas web devem se comunicar. No caso do

JSF esses sistemas podem se imaginar como sistema cliente em *Javascript* e outro sistema servidor em *Java*. Um esforço em particular em formato de pesquisa de doutorado foi capaz de unir grande parte dos sistemas à implementar a mesma solução. O esforço foi feito por Roy Fielding (cientista da computação influente na criação do HTTP 1.1) para implementar um conjunto de princípios para que serviços web se integrem de maneira uniforme e opaca, esse conjunto de princípios foi denominado como REST.

4.2 REST (REpresentational State Transfer)

De acordo com Stefan Nilkov (2010) REST é definido como um conjunto de princípios que definem como padrões web (*HTTPs* e *URIs*) devam ser supostamente utilizados (bastante diferente da maneira implementada por várias pessoas). A promessa é que se você aderir aos princípios do REST enquanto projeta sua aplicação, você terá um sistema que aproveitará a arquitetura da *Web* já pronta ao seu benefício. Em resumo, os cinco princípios chaves são:

- Dê a qualquer recurso um *ID* (Identificador).
- Conecte os recursos com *links* (o mesmo do HTML) através de referência aos *IDs*.
- Use os métodos padrão (GET, POST, PUT, DELETE).
- Recursos podem ter múltiplas representações (XML, JSON, FormatoArbitrário).
- Comunique-se sem compartilhar estado (Importante para fazer *cache*, se uma chamada pode ser respondida exatamente como a anterior, então teremos um cenário ideal para fazer *cache*).

Diante desses princípios, veremos a seguir como um stack feito puramente em Javascript é concebido.

4.3 MEAN (MongoDB, Express, Angular, Node.js)

MEAN de acordo com seu criador Valeri Karpov (2013) é um conjunto de tecnologias construída em cima de softwares escritos em Javascript, popular em 2014 e é alternativa para a plataforma XAMPP (Apache, MySQL, PHP, Linux). XAMPP de acordo com os criadores da plataforma (Kai Seidler, Kay Vogelgesang) é uma plataforma popular, simples e direta de criar sites, pois mesmo o Java sendo a linguagem mais popular entre empresas, ela não deixa de ser uma linguagem de alta complexidade a qual assusta os desenvolvedores iniciantes.

A partir da entrada do MEAN a regra de negócio passa a ser tratada no cliente com o *AngularJS* (framework cliente *Javascript*), com essa mudança os servidores passam a ter menos código, servindo apenas como detentor da informação e o cliente a agregar mais código. Dessa forma mais profissionais que dominam o *Javascript* fiquem mais independentes de outras tecnologias, consequentemente fazendo com que a linguagem suba várias posições no mercado. Com a popularização do *Javascript* o *node.js* acaba por se tornar uma opção viável pois há bastante mercado para ser contratado, além da natureza assíncrona do ambiente que favorece bastante servidores web, que não precisam fazer muitas operações pesadas, mas sim requisições externas ao mesmo tempo.

Com essas tecnologias a construção de banco de dados é a única parte que o Javascript não cobre, fazendo com que o MongoDB surja como uma opção forte. O MongoDB foi construído para ser utilizado nativamente em Javascript e utiliza o

formato *NoSQL* (NãoSQL). Com o MongoDB o ecossistema Javascript é fechado e o desenvolvedor só precisa ter domínio de uma linguagem.

O MEAN não utiliza o padrão REST em nenhum momento, pois as ferramentas são integradas por escolha do programador. Caso o desenvolvedor escolha por utilizar o REST a aplicação ganha toda a robustez elucidada no subcapítulo 4.1 e fica manutenível a qualquer por quem for dar suporte. O sistema utilizaria o Express para declarar os recursos no servidor da seguinte maneira.

```
var express = require('express');
var router = express.Router();
var app = express();

router.get('/users', listUsers);
router.post('/users', createUser);
router.get('/users/1', getUser);
router.put('/users/1', updateUser);
router.delete('/users/1', deleteUser);

app.listen(3000, function (port) {
  console.log('Escutando na porta', port);
});
```

Figure 4: Código Express.
Fonte: Próprio Autor

Nesse pequeno exemplo de código percebe-se a simplicidade de implementar um recurso em REST, pois o Express utiliza os mesmos verbos HTTP como métodos de roteamento. Com esse servidor rodando o cliente pode pedir por usuários através do Angular utilizando qualquer serviço HTTP de pedido.

O grande problema do MEAN *stack* é a grande quantidade de conceitos que o desenvolvedor tem de aprender antes de utilizar cada tecnologia, além do domínio natural do REST que o desenvolvedor precisa ter. O *ExpressJS* é a tecnologia que utiliza o menor número de conceitos, mesmo assim uma aplicação de médio porte precisa utilizar de vários *middlewares* (serviços que interceptam a requisição e adicionam funcionalidades) como autenticação, segurança, auditoria, etc. Já no Angular existe uma grande gama de conceitos que a figura a seguir elucidada, mesmo assim o framework ainda possui mais conceitos que não estão contidos na foto, como *providers*, *controller as scope*, etc.

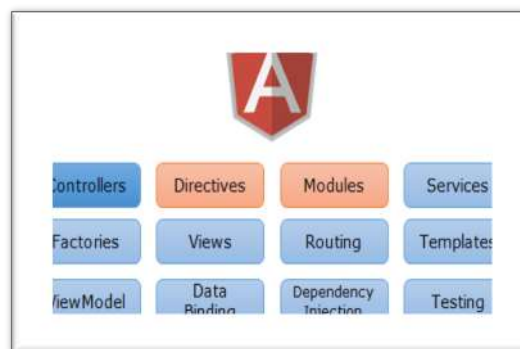


Figure 5: Conceitos de AngularJS.

Fonte: <https://codewala.net/2014/06/03/learning-angularjs-with-examples-part-2/> acesso em 04/09/2016

5 DISTRIBUTED DATA PROTOCOL

O DDP de acordo com Sashko Stubailo na definição da especificação do DDP é um protocolo responsável pela comunicação cliente-servidor que suporta duas operações:

- *Remote Procedure Call* (Chamada de Procedimento Remota) do cliente para o servidor.
- Uma lista de documentos que são assinados pelo cliente, com o servidor responsável por manter essa lista de informações atualizadas.

Este protocolo funciona em cima de um protocolo *Websockets* caso do *browser* tendo suporte à essa tecnologia, caso não, uma simulação de eventos é feita através da biblioteca *Sock.js*. Esse protocolo é a base de funcionalidade do DDP, pois sem ele, não há como o servidor enviar informações para o cliente, através das assinaturas de documentos.

5.1 WEBSOCKETS

O RFC 6455 mais conhecido como o protocolo *WebSocket* tem o propósito de estabelecer uma comunicação bidirecional em cima de uma só conexão HTTP, algo que antes era simulado através de técnicas rudimentares de sincronia de informação. Esse protocolo é implementado utilizando o HTTP como camada de transporte para aproveitar de todas as vantagens que o HTTP oferece (*proxy*, autenticação, filtros). Com essa característica, o *WebSocket* funciona em cima das portas padrões HTTP (80, 443). Mesmo tendo alta compatibilidade com o HTTP, o *WebSocket* pode ser implementado em qualquer tecnologia que utilize um método de *HandShake* simples.

De acordo com a *Microsoft*, o *Handshake* do Protocolo de Segurança da camada de Transporte (*Transport Layer Security*, TLS) é responsável pela autenticação e troca de chaves necessárias para se estabelecer ou se resumir uma sessão, O *WebSocket* utiliza do mesmo mecanismo para se conectar do HTTP para o *WebSocket*, esse mecanismo é chamado de *Upgrade Request*.

O protocolo DDP foi construído em cima de *Websockets* como base pois é bidirecional por natureza, adicionando uma padronização de mensagens em cima a fim de estabelecer uma maneira de clientes e servidores serem desenvolvidos com a mesma capacidade de interação que o *Meteor* (plataforma que o utiliza) oferece. Os subcapítulos adiante foram retirados e traduzidos diretamente da especificação do DDP.

5.2 ESTABELECENDO UMA CONEXÃO DDP

As mensagens trocadas por aplicações e servidores que implementam o protocolo DDP possuem uma estrutura padronizada a fim de manter todos os clientes capazes de conversar em uma semântica rica com o servidor, as mensagens são expressas na seguinte árvore JSON (conteúdo extraído e traduzido direto da especificação):

5.2.1 Mensagens

- *connect* (cliente -> servidor)
 - *session* : *string* (só existe se estiver tentando conectar em uma sessão já existente, no caso de uma reconexão)
 - *version* : *string* (a versão do protocolo estabelecido)
 - *support* : *Array<String>* (versões do protocolo que o cliente sabe conversar, em ordem de preferência)

- *connected* (servidor -> cliente)
 - *session* : *string* (um identificador da sessão DDP)
- *failed* (servidor -> cliente)
 - *version* : *string* (uma versão de protocolo sugerida para se conectar)

5.2.2 Procedimento

O servidor pode mandar uma mensagem inicial em formato objeto JSON sem a chave *'msg'*. Se for assim, o cliente deve ignorá-la. O cliente não deve esperar por esta mensagem (Esta mensagem era uma forma de implementar a atualização automática do Meteor, agora é incluída apenas para forçar clientes antigos a atualizar).

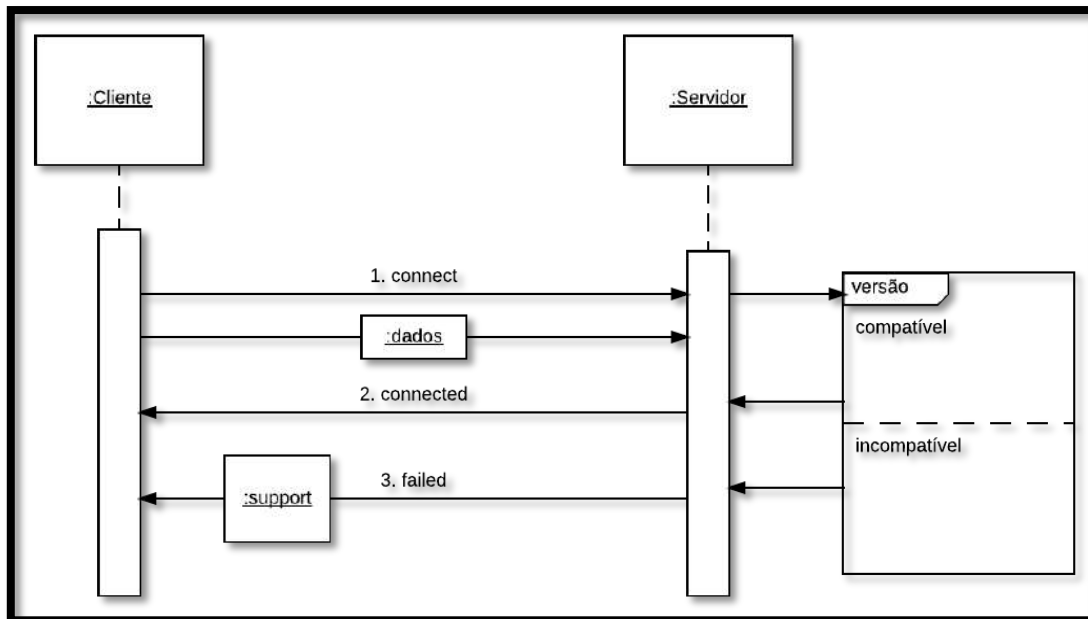


Figure 6: Mensagem de conexão.
Fonte: Próprio Autor.

1. O cliente manda uma mensagem “*connect*”
2. Se o servidor estiver disponível para conectar na versão estabelecida pelo cliente através do conteúdo da mensagem “*connect*”, mandará uma mensagem “*connected*” de volta.
3. Caso este não seja o caso, o servidor mandará uma mensagem “*failed*” com a versão disponível para conversação, através do campo “*support*” e fechará a devida requisição.
4. O cliente estará livre então para se reconectar através de uma versão diferente do protocolo DDP. O cliente pode otimistamente mandar novas mensagens depois da mensagem “*connect*”, assumindo que o servidor irá suportar o protocolo proposto. Se o servidor não suportar aquela versão, as mensagens adicionais serão ignoradas.

As versões no campo “*support*” da mensagem “*connect*” são ordenadas de acordo com a preferência do cliente. Se, de acordo com esta ordenação, o protocolo não estiver na melhor versão que o servidor suporta, o servidor deverá forçar o cliente a trocar para a melhor versão através da mensagem “*failed*”.

Quando um cliente estiver conectando a um servidor pela primeira vez ele deverá tipicamente setar a “*version*” para a versão mais preferida. Se desejado, o cliente pode então lembrar da versão que foi ultimamente negociada com o servidor e então começar com aquela versão em conexões futuras. O cliente poderá depender do servidor

enviar uma mensagem “*failed*” se uma versão atualizada estiver disponível em detrimento de uma atualização do cliente ou do servidor.

5.2.3 Heartbeats (Batidas de Coração)

- *ping* (cliente ou servidor -> servidor ou cliente)
 - *id* : *string* opcional (id usado para relacionar com a resposta)
- *pong* (cliente ou servidor -> servidor ou cliente)
 - *id* : *string* opcional (mesmo id recebido na mensagem “ping”)

5.2.4 Procedimento

Em qualquer momento depois da conexão ser estabelecida qualquer lado pode enviar uma mensagem “*ping*”. Quando o outro lado receber uma mensagem “*ping*” deverá imediatamente responder com uma mensagem “*pong*”. Se a mensagem “*ping*” incluir um campo “*id*”, a mensagem “*pong*” deverá conter o mesmo campo “*id*”.

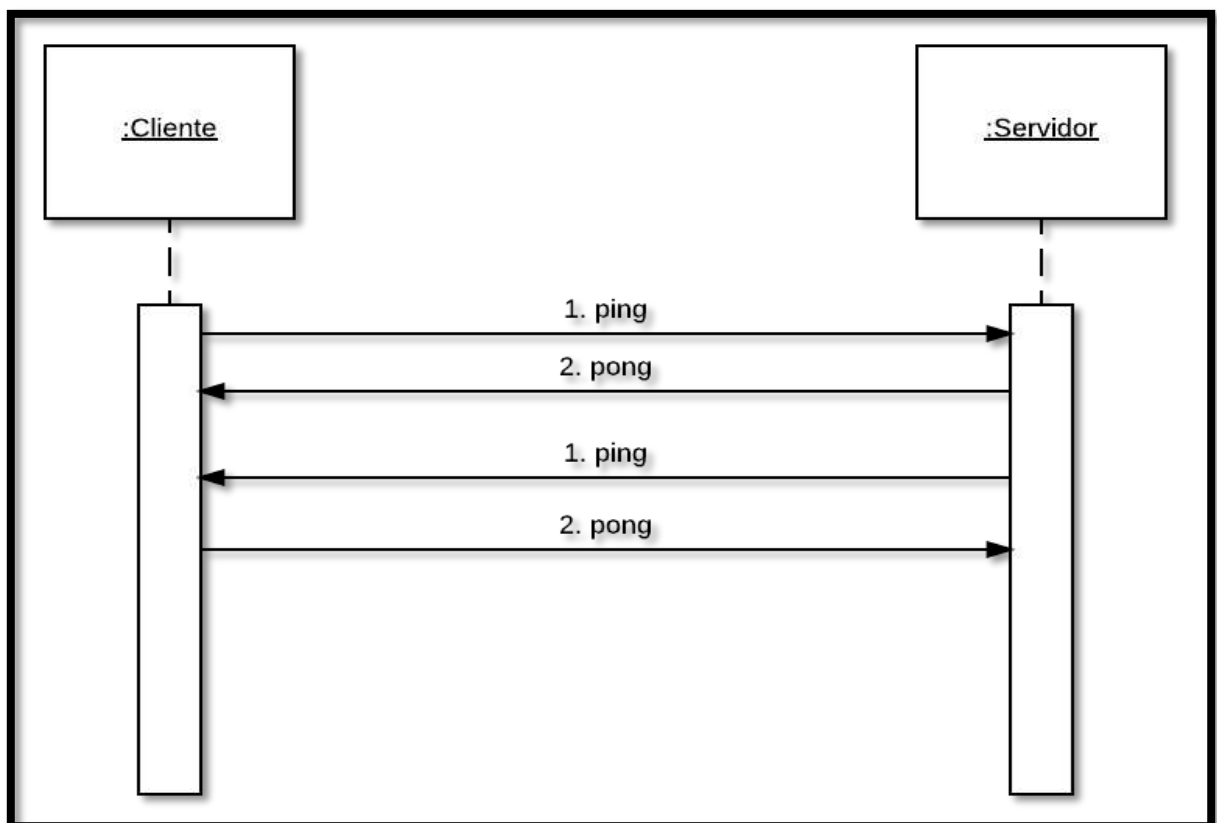


Figure 7: Mensagens de checagem de conexão.
Fonte: Próprio autor.

5.3 MANUSEANDO DADOS

Estabelecer uma conexão é o passo inicial para a conexão cliente servidor, depois da conexão estabelecida, o protocolo passa a conversar então com coleções Mongo (termo utilizado como documentos em um banco *NoSQL*, similar a uma tabela *SQL*), é nesse passo que a semântica é mais importante, pois define o que a aplicação cliente pode fazer com o servidor.

5.3.1 Mensagens

- *sub* (cliente -> servidor)
 - *id* : *string* (um *id* definido arbitrariamente pelo cliente para essa assinatura)

- *name* : *string* (o nome da assinatura)
- *params* : *array* opcional *JSON* (parâmetros da assinatura)
- *unsub* (cliente -> servidor)
 - *id* : *string* (o *id* passado para o “*sub*”)
- *nosub* (servidor -> cliente)
 - *id* : *string* (o *id* passado para o “*sub*”)
 - *error* : Erro opcional (um erro passado pela assinatura quando concluída, ou *sub-not-found*)
- *added* (servidor -> cliente)
 - *collection* : *string* (nome da coleção)
 - *id* : *string* (*ID* do documento)
 - *fields* : objeto *JSON* opcional com os valores adicionados
- *changed* (servidor -> cliente)
 - *collection* : *string* (nome da coleção)
 - *id* : *string* (*ID* do documento)
 - *fields* : objeto opcional com os valores *JSON*
 - *cleared* : *array* opcional de *strings* (nome dos campos a serem eliminados)
- *removed* (servidor -> cliente)
 - *collection* : *string* (nome da coleção)
 - *id* : *string* (*ID* do documento)
- *ready* (servidor -> cliente)
 - *subs* : *array* de *strings* (ids passados para o “*sub*” que foram enviados na primeira leva de dados)
- *addedBefore* (servidor -> cliente)
 - *collection* : *string* (nome da coleção)
 - *id* : *string* (*ID* do documento)
 - *fields* : objeto *JSON* opcional com os valores
 - *before* : *string* ou nulo (o *ID* do documento para ser adicionado atrás, ou nulo se for ao fim da lista)
- *movedBefore* (servidor -> cliente)
 - *collection* : *string*
 - *id* : *string* (o *ID* do documento)
 - *before* : *string* ou *null* (o *ID* do documento para ser movido atrás, ou nulo se for para o fim da lista)

5.3.2 Procedimento

O cliente especifica partes da informação que está interessado através do envio da mensagem “sub” para o servidor. A qualquer momento, mas geralmente informado pela mensagem “sub”, o servidor pode enviar mensagens com dados para o cliente. Os dados consistem em mensagens “added”, “changed” e “removed”, como demonstrado na figura 8. Essas mensagens modelam um set de coleções no cliente ao qual ele deve manter atualizado.

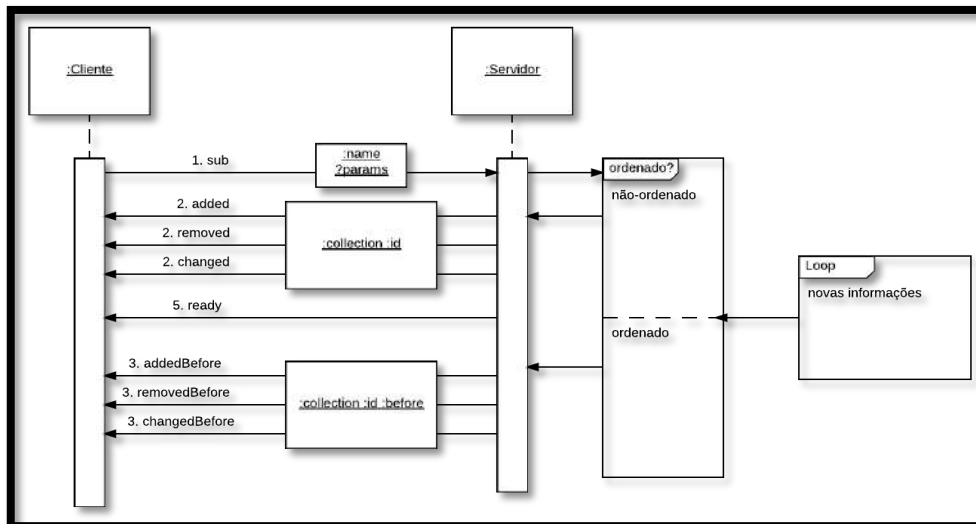


Figure 8: Procedimento de assinatura e publicação entre cliente e servidor.
Fonte: Próprio autor.

- Uma mensagem “added” indica que um documento foi criado na coleção local. O ID do documento é especificado no campo “id”, e os campos dos documentos são especificados no objeto “fields”, se presente, indicando os campos do documento ao qual deva ser substituído pelos novos valores. O campo “cleared” contém a lista de valores que não estão mais contidos no documento.
- Uma mensagem “removed” indica que um documento foi removido da coleção local. O campo “id” indica o ID do documento.
- A coleção é ordenada ou não. Se ordenada, a mensagem “added” será substituída pela “addedBefore”, a qual adicionalmente contém o ID do documento após daquele que está sendo adicionado. Se o documento está sendo adicionado no fim, “before” é setado para null. Para uma coleção dada, o servidor deve somente enviar mensagens “added” ou “addedBefore”, nenhuma das duas intercaladas, e não deve mandar “movedBefore” para uma coleção com “addedBefore”.
- O cliente mantém uma lista de dados por coleção. Cada assinatura não pode ter sua própria lista, podendo assim haver sobreposição fazendo com que o servidor envie apenas a união dos fatos sobre uma coleção de dados. Por exemplo, se a assinatura A diz que o documento x tem os campos {foo: 1, bar: 2} e a assinatura B diz que o documento x tem os campos {foo: 1, baz: 3}, então o cliente será informado que o documento x tem os campos {foo: 1, bar: 2, baz: 3}. Se os valores do campo de assinaturas diferentes conflitarem, o servidor deve mandar só um dos possíveis valores.
- Quando uma ou mais assinaturas terminarem de enviar seu pacote inicial de dados, o servidor deve enviar uma mensagem “ready” com seus Ids.

5.4 REMOTE PROCEDURE CALLS

O protocolo DDP utiliza mensagens estilo RPC para enviar alterações para o servidor, caso não houvesse essa característica, a aplicação seria incapaz de fazer mudanças no sistema. A seguir será mostrada as estruturas das mensagens que o cliente envia e o servidor responde.

- *method* (cliente -> servidor)
 - *method* : string (nome do método)
 - *params* : array JSON opcional (parâmetros do método)

- *id* : *string* (um *ID* arbitrário determinado pelo cliente para identificar a chamada)
- *randomSeed* : *JSON* opcional (uma *seed* arbitrária determinada pelo cliente para identificar geradores pseudo-aleatórios)
- *result* (servidor -> cliente)
 - *id* : *string* (o id passado para o “*method*”)
 - *error* : erro opcional (um erro lançado pelo método (ou método-não-encontrado))
 - *result* : *JSON* opcional (o valor do retorno do método, se houver algum)
- *updated* (servidor -> cliente)
 - *methods* : *array* de *strings* (ids passados pelo “*method*”, todos aqueles que foram refletidos nos dados da mensagem)

5.4.1 Procedimento

O cliente manda uma mensagem do tipo “*method*” para o servidor, assim o servidor responde com uma mensagem “*result*” para o cliente, carregando o resultado do método ou uma mensagem de erro apropriada. Chamadas de método podem afetar dados que o cliente pode ter assinado. Assim que o servidor tiver terminado de enviar para o cliente todas as mensagens de dados relevantes baseada nessa chamada de procedimento, o servidor deve enviar uma mensagem “*updated*” para o cliente com os *IDs* desses métodos. Não existe ordem particular entre mensagens “*result*” e “*updated*” para chamada de método.

O cliente pode enviar um valor *JSON randomSeed*. Se provido, esse valor é usado para alimentar a criação de números pseudorrandômicos. Se usado a mesma *seed* com o mesmo algoritmo, os mesmos valores pseudorrandômicos serão gerados tanto no cliente quanto no servidor. Em particular, isso é usado para gerar ids para documentos recém-criados. Se *randomSeed* não for provido, então os valores gerados no servidor e no cliente não serão idênticos.

Atualmente *randomSeed* é esperado que seja uma *string*, e o algoritmo ao qual os valores são produzidos não está ainda documentado. É provável que futuramente ele será formalizado quando estiver claro quais são os requisitos, ou quando uma implementação compatível precisar que isso esteja implementado.

5.5 ERROS

Erros aparecem em mensagens “*result*” e “*nosub*” no campo opcional “*error*”. Um erro é um objeto com os seguintes campos:

- *error* : *string*
- *reason* : *string* opcional
- *details* : *string* opcional

Esses erros são usados para representar erros levantados pelo método ou assinatura, ao mesmo que uma tentativa de inscrever a uma assinatura desconhecida ou chamar um método não existente.

Outras mensagens errôneas mandadas pelo cliente para o servidor podem resultar no recebimento de uma mensagem *top-level* “*msg: ‘error’*” na resposta. Essas condições incluem:

- envio de mensagens que não são *JSON* válidos
- tipo desconhecido de “*msg*”
- outro tipo de requisição malformada do cliente (não incluindo campos obrigatórios)
- envio de qualquer outro tipo de mensagem que não seja “*connect*” como primeiro envio, ou envio de “*connect*” como segundo envio ou posterior.

A mensagem de erro contém os seguintes campos:

- *reason* : *string* descrevendo o erro
- *offendingMessage* : se a mensagem original foi parseada corretamente, será enviada por aqui

5.6 APLICAÇÃO EM DDP (Meteor)

De acordo com Matt Debergalis, *Meteor* é uma plataforma que trouxe novos conceitos focando no que torna uma aplicação mais eficiente na web, dentre todas as tecnologias que foram estudadas, nenhuma focou no conceito que mais dominou a internet, que é a comunicação (que é a função do *HTTP*). A mensagem, e a forma como ela é passada e percebida, não se tem tido o foco necessário durante a história da programação distribuída.

De acordo com Alan Kay (1998), cientista proeminente da *Xerox* e *Apple* enfatiza:

Se você focar apenas na mensagem - e perceber que um bom metassistema pode ser ligado a várias arquiteturas de segundo nível usadas em objetos - então muito da linguagem-, Interface do Usuário-, e discussões na comunidade fica bastante sujeita a controvérsia. (tradução do autor).

Com essa idéia, o protocolo *DDP* foi concebido, que para suportar uma aplicação Meteor, precisa-se de um ambiente que converse em *DDP* em cima de *WebSockets*, a aplicação se torna viva em relação ao *backend*, qualquer atualização no sistema automaticamente é refletida em todos os clientes que tem um vínculo com o sistema. Tendo em vista esse desafio, foi feito um trabalho fundamental na maneira de se construir uma aplicação, muito mais parecida com aquela que se vinha trabalhando no ambiente nativo, dentre esse trabalho, foram construídas várias tecnologias para dar razão a esses conceitos e generalizar todas as possibilidades de construção de sistemas, dentre elas seguem, *Blaze*, *Tracker*, *IsoBuild*, etc. Todas essas tecnologias são evidenciadas nas duas figuras 9 e 10 a seguir:

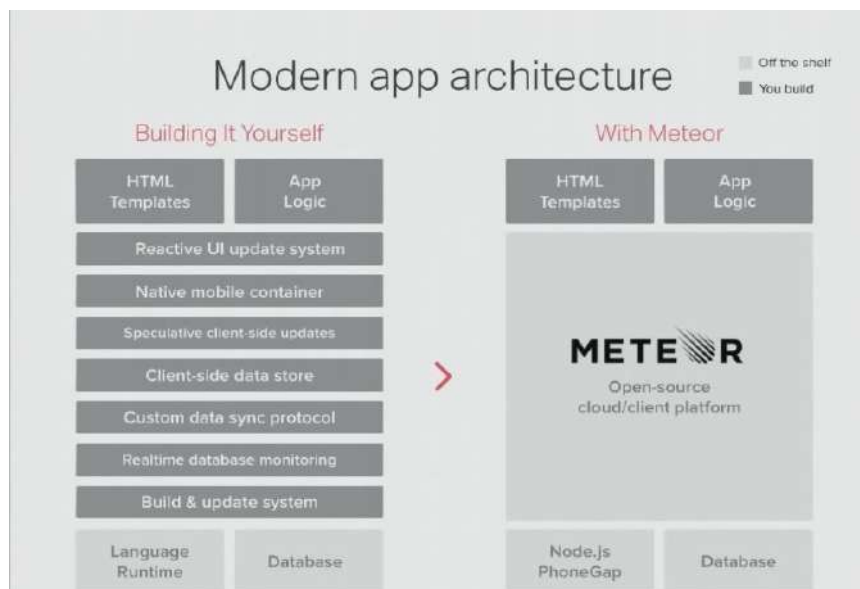


Figure 9: Arquitetura de uma Aplicação Web.

Fonte: <https://youtu.be/x4nxDzslE4?t=10m19s> acesso em 04/09/2016

Com a ajuda dessas tecnologias, a integração do sistema fica para responsabilidade da plataforma, deixando o programador muito mais produtivo e focado

no produto em que ele deseja entregar. Com toda a plataforma ao dispor, o desenvolvedor acaba por se deparar com uma forma de programação chamada reativa, ou seja, sem a necessidade de notificar o sistema de nenhuma mudança.

A programação reativa é um dos princípios da programação funcional, e é um conceito recente que é bastante trabalhado na comunidade de programadores, pois com ele o LoC (*Lines of Code*, ou linhas de código) caiu drasticamente, tornando sistemas previamente complicados e complexos em sistemas pequenos, pois a maioria das regras era para a atualização e notificações de novos recursos no banco.

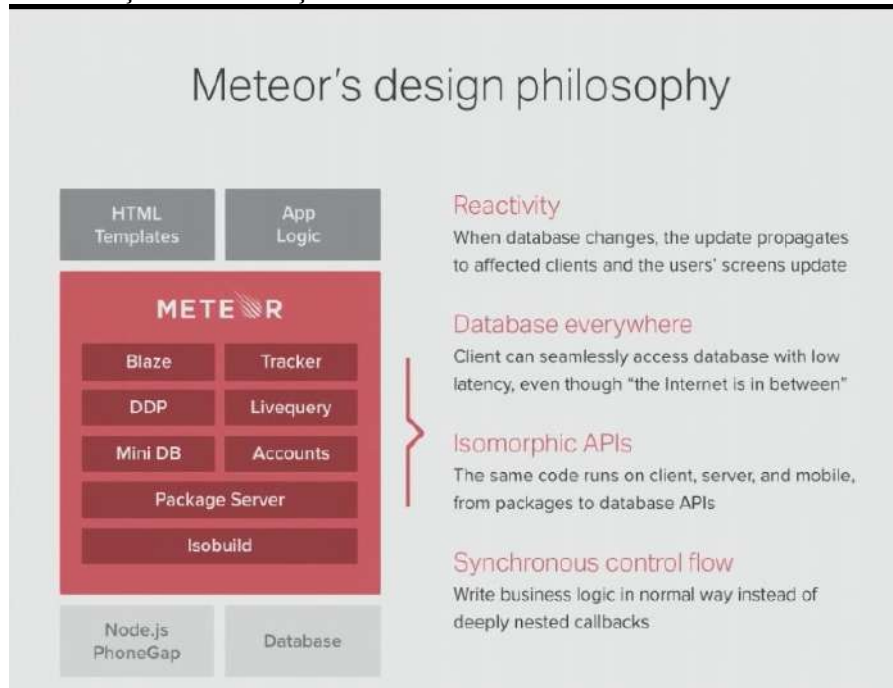


Figure 10: Filosofia do MeteorJS.

Fonte: <https://youtu.be/x4nxDzslE4?t=11m41s>, acesso em: 04/09/2016

Com a programação reativa um conceito foi trazido das linguagens funcionais, chamada programação declarativa. Um exemplo de linguagem de programação declarativa é o *SQL*. De acordo com esse conceito as linguagens de *template* ficaram muito mais próximas da interação com o banco de dados, pois o banco é trazido para o *template* como será demonstrado no código.

6. COMPARATIVO ENTRE DDP e HTTP

Neste capítulo serão abordados comparativos em forma de relatórios entre os dois protocolos *HTTP* com *REST* e *DDP*, neste mesmo esquema será explicado que não só a performance é aumentada como a própria concepção das mensagens é sentida pelo desenvolvedor. Na concepção do *HTTP*, há a requisição e a resposta, já no *DDP* a assinatura e a publicação. Esse esquema vem sendo cada vez mais expandido na web através de novos esquemas de banco de dados (Kleppmann M., 2015. *Turning the Database Inside Out*), e sistemas distribuídos (*Apache Kafka* e *Samza*). Então se tem em mente que a performance pode ser tratada como uma característica inerente a mais além dessas outras concebidas.

Quando se pesquisa relatórios já abordados na área de protocolos, é de conhecimento comum a prática de testes de carga no servidor, testes que provam claramente a performance de um protocolo de comunicação de servidor mas deixam a desejar quando o foco passa para o cliente, pois o próprio termo jamais foi entendido como o browser e sim a interface em formato de *template HTML*. É neste foco que este relatório ficará abordado.

Para a execução dos testes com o DDP, o sistema está disponível na plataforma *MeteorJS*, e para a simulação de um ambiente de execução foi adicionado uma cláusula *production* para o funcionamento, sendo assim o comando digitado *meteor -production*, nesse cenário todo o sistema está na configuração apropriada para testes de performance, pois ele retira todos os pacotes que facilitam o desenvolvimento como *debuggers* a fim de diminuir a carga na requisição.

Para comparar o DDP com Meteor, uma aplicação muito próxima é programada utilizando o framework *Ruby on Rails*, afim de simular o cenário mais preciso para testes. A proximidade vem da incapacidade do HTTP de ter as mesmas funcionalidades que a aplicação necessita, então alguns componentes foram alterados, ex: O botão de busca foi adicionado no lugar do adicionar usuário, visto que uma aplicação HTTP não tem condições de reconhecer eventos de digitação de um teclado afim de gerar uma busca assíncrona, somente com o uso de AJAX. A aplicação possui a mesma regra de negócio, mas não tem a mesma funcionalidade interativa que a construída com *Meteor*, pois só com o uso excessivo de bibliotecas (*RubyGems*, repositório de bibliotecas do *Rails*) o *Rails* se torna mais interativo tão quão o Meteor que já é por natureza. Não se têm uso de *XHR* nas telas para evitar muita complexidade no sistema, já que o *XHR* utiliza o *Javascript* constantemente. HTTP com o REST utilizado da forma mais ideal, sem compartilhar nenhuma informação com o cliente, pois toda a regra está no servidor.

Com as aplicações prontas, segue os testes e ferramenta para o comparativo. A ferramenta de teste para servidores utilizada está disponível em código aberto no *GitHub*, essa ferramenta é:

- *Siege*, fonte: <https://github.com/joedog/siege>

A ferramenta *Siege* é utilizada como força bruta para medir a resiliência e robustez do sistema, pois retorna um relatório de performance com estatísticas, dentre elas, número de transações, disponibilidade, dados transferidos, tempo de resposta, etc.

Os relatórios de servidor serão menos conclusivos do que os de clientes, pois não coincide no foco da intenção do uso do *Websocket*, que significa manter uma conexão aberta com o cliente, esses pacotes sempre serão isolados e não poderão levar vantagem do DDP.

Com essa ferramenta os servidores terão de provar resiliência e robustez para casos de uso constante de requisição, o servidor deve ser atacado constantemente por vários usuários paralelos afim de simular um cenário real de crescimento de uso. Nesse ponto será testada sua robustez.

6.1 ROBUSTEZ

De acordo com N. Stricker (2014) a definição de robustez e de sistemas robustos de produção vem de como um sistema que consiga lidar com distúrbios a fim de manter o desempenho em um alto nível. Isso pode tanto ser feito sendo resistente a distúrbios (resiliência, agilidade) ou tendo uma reação apropriada à diferentes condições (flexibilidade, mudança), diante desses dois cenários um ataque de DDOS (*Distributed*

Denial of Service) é escolhida como a forma de convergir esses cenários em um tipo de ataque.

A aplicação sofrerá diversos ataques de clientes fazendo pedidos, não compete a esse teste a capacidade computacional do sistema, e sim a capacidade de lidar com entrada e saída simultânea.

Os dois sistemas serão atacados com requisições de forma igual para cada servidor, isto é, no caso do HTTP serão feitas requisições com pacotes únicos sem sessão entre eles, mantendo a conexão o máximo possível sem-estado (*stateless*), pois é como foi planejado o HTTP.

Pode-se perceber na figura 11 que os testes em *Ruby on Rails* a existência de falha nas transações com o servidor, denunciando uma fadiga no sistema, o campo *Longest Transaction* (Transação mais longa) também mostra um intervalo de tempo bastante insuportável para o usuário (12,86s). Isso mostra que tanto Ruby não é apropriado para servidores com alta taxa de concorrência quanto o Rails não prevê uma carga grande no sistema.

```
Lifting the server siege...      done.
Transactions:                  497 hits
Availability:                  91.70 %
Elapsed time:                  348.28 secs
Data transferred:             4.15 MB
Response time:                 9.73 secs
Transaction rate:              1.43 trans/sec
Throughput:                    0.01 MB/sec
Concurrency:                   13.89
Successful transactions:       497
Failed transactions:           45
Longest transaction:           12.86
Shortest transaction:          4.20
```

Figure 11: Teste no Rails.
Fonte: Próprio Autor.

Na figura 12 vemos outro cenário, contrário ao demonstrado anterior. O servidor *Meteor* feito em *Node* lida com transações paralelas com uma facilidade notável, no campo *Transaction rate* (Taxa de transação) percebe-se a maior diferença, enquanto o Rails está próximo de uma transação por segundo, o *Node* mostra um número próximo de 30, ainda mais não havendo nenhuma falha durante o teste.


```

Lifting the server siege...      done.

Transactions:                   6843 hits
Availability:                   100.00 %
Elapsed time:                   234.12 secs
Data transferred:              3.73 MB
Response time:                 0.01 secs
Transaction rate:              29.23 trans/sec
Throughput:                    0.02 MB/sec
Concurrency:                   0.29
Successful transactions:       6843
Failed transactions:           0
Longest transaction:           0.14
Shortest transaction:          0.00

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.

```

Figure 12: Teste no Meteor.
 Fonte: Próprio Autor.

A importância nesses testes de resistência está em mostrar a disponibilidade dos dois serviços, na demonstração que mesmo o servidor Meteor usando um servidor não adaptado para a carga constante de requisições HTTP, ele está mais preparado para testes duros pois tanto nas requisições quanto as respostas são menores pois ele não precisa passar tanta informação para o cliente, essa afirmação será abordada no próximo capítulo em que se vê no navegador o que de fato está sendo recebido pelo servidor. Este teste no lado do cliente está subdividido em requisição inicial, requisição através da navegação e a quantidade de informação em *KB/s* que se é passada entre todo o processo.

6.2 REQUISIÇÃO INICIAL

O teste de requisição inicial das figuras 13 e 14 demonstra como os sistemas respondem à primeira requisição, pode-se perceber somente no tamanho do arquivo “localhost” onde se encontra o a marcação excesso de informação do *Rails* em relação ao *Meteor*, essa informação está somente em texto HTML que o servidor envia com a alteração da tela, já no *Meteor*, o cliente já entende um nível abaixo de *Tags*, já que está lidando direto com o *DOM* (Document Object Model), nesse caso, o que vem do servidor é apenas a informação necessária para que o cliente manipule o *DOM*.

<input type="checkbox"/>	websocket	101	websoc..	VM784:35	0 B	Pending
<input type="checkbox"/>	localhost	200	docum...	Other	783 B	312 ms
<input type="checkbox"/>	2c5db8d7c3228e255c5cad58d62f5d2f03f66fbf.css?meteor_css_r...	304	styleshe...	(index):4	203 B	34 ms
<input type="checkbox"/>	5fe74cc4234274e65f8a0a2a1f26f87e0cc28396.js?meteor_js_reso...	304	script	(index):14	203 B	346 ms
<input type="checkbox"/>	favicon.ico	200	vnd.mic...	Other	(from di...	2 ms
<input checked="" type="checkbox"/>	brasao2.gif	200	gif	Other	(from di...	8 ms
<input type="checkbox"/>	info?cb=b_9t8ls94z	200	xhr	5fe74cc...js?me...	368 B	12 ms
<input type="checkbox"/>	glyphicons-halflings-regular.woff2	304	font	5fe74cc...js?me...	196 B	8 ms

Figure 13: Requisição Inicial do Meteor.
 Fonte: Próprio Autor.

Name	Status	Type	Initiator	Size	Time
localhost	200	docum...	Other	26.4 KB	608 ms
application.self-a13b6e48deaf39fa578744539ed7e9b3b0e29c6d...	304	styleshe...	(index):6	326 B	231 ms
jquery.self-bd7ddd393353a8d2480a622e80342adf488fb6006d6...	304	script	(index):7	326 B	118 ms
jquery_ujs.self-784a997f6726036b1993eb2217c9cb558e1cbb80...	304	script	(index):8	326 B	275 ms

Figure 14: Requisição Inicial do Rails.
Fonte: Próprio autor.

6.3 REQUISIÇÕES ATRAVÉS DA NAVEGAÇÃO

O teste de requisição na figura 15 demonstra como o sistema se comporta através de cada interação com o usuário, isto é, um clique em uma busca, uma troca de tela, e uma adição de visitante ao sistema, no total não se chega a 1Kb de dados transferidos durante todo o processo (notar que há um processo de auditoria em ambos os projetos afim de demonstrar relatórios de entrada e saída de visitantes no prédio para o governante haver ciência).

```

Message Stack Trace
root: {} 4 keys
  msg: "added"
  collection: "history"
  id: "FGJhZr2Q5wf4axhQP"
  fields: {} 5 keys
updated {"msg":"updated","methods":["8"]} 33 B
  
```

Figure 15: Mensagens DDP do cliente em JSON.
Fonte: Próprio Autor.

No exemplo da figura 16 do *Rails* com HTTP podemos perceber outra vez um excesso relativo de informação, pois como o HTTP não tem ideia das partes que se repetem, o servidor envia todo o *template* com a mudança, resultando em 26kb de transferência para uma ação.

Name	Status	Type	Initiator	Size	Time
2	302	text/html	Other	870 B	377 ms
localhost	200	docum...	http://localhost...	26.4 KB	367 ms
application.self-a13b6e48deaf39fa578744539ed7e9b3b0e29c6d...	200	styleshe...	(index):6	(from di...	5 ms
jquery.self-bd7ddd393353a8d2480a622e80342adf488fb6006d6...	200	script	(index):7	(from di...	9 ms
jquery_ujs.self-784a997f6726036b1993eb2217c9cb558e1cbb80...	304	script	(index):8	(from m...	0 ms
tether.self-49a614f96d3d9b228d4f800376a8db0a8315c1f10eb7...	304	script	(index):9	(from m...	0 ms

Figure 16: Mensagem HTTP em HTML.
Fonte: Próprio autor.

6.4 TEMPO DA INFORMAÇÃO ATÉ A UTILIZAÇÃO

No relatório da figura 17 os sistemas simulam um processo de inicialização de tela com a informação necessária para o uso do sistema e o tempo necessário para se ter uma navegação. A ferramenta utilizada é a aba “*Timeline*” (Linha do tempo) do *Chrome*, essa ferramenta é capaz de medir o tempo necessário em que a página se torna navegável, além disso, ela avalia uma vasta gama de recursos com gráficos. Nesse caso

é utilizado os recursos “*Network*”, “*JS Profile*”, “*Memory*” e “*Paint*”. Cada recurso será uma linha de cor específica no relatório. O sistema será testado a partir do momento da requisição inicial.



Figure 17: Quantidade de informação inicial em meteor plano.
Fonte: Próprio autor.

Neste gráfico pode-se perceber uma alta utilização do sistema no momento inicial, caso ideal para uma aplicação web, pois significa que o servidor está servindo os recursos com presteza, todo o processo ocorre até a janela de 1 segundo, ao qual o processo do Javascript acaba, nesse momento o browser emite o evento de estar pronto, no qual é nessa hora que o resto dos *scripts* são lançados, percebe-se uma atividade estável no plano azul embaixo, sinal de que a aplicação não está com nenhum gargalo, diferente da figura a seguir, que demonstra como o Rails se comporta na requisição inicial.

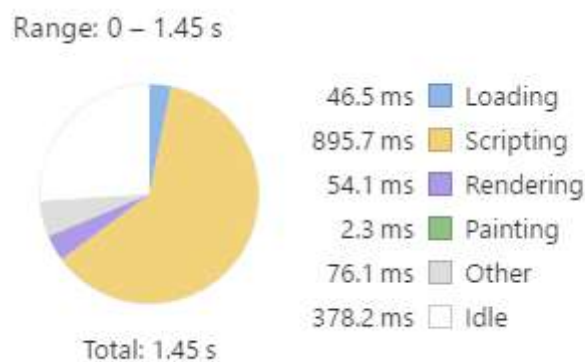


Figure 18: Quantidade de informação inicial em pizza para o Meteor.
Fonte: Próprio autor.

Na figura 18 é demonstrado um caso quase perfeito de utilização dos recursos do browser, a pintura é praticamente instantânea enquanto que a renderização um pouco mais lenta, a alta leitura de script é um sinal de que a aplicação está sendo montada no lado do cliente, e a parte branca representa o tempo em que o browser não exerceu atividade, este é um cenário comum pois alguns javascripts se utilizam de momentos em que o DOM emite um evento “*ready*” para só assim ser trabalhado por certos scripts, sendo esse o caso. Tudo ocorre em uma janela de 1 segundo e meio, a partir do segundo 0.

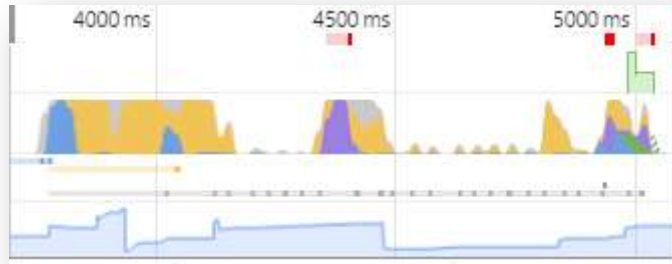


Figure 19: Quantidade de informação inicial em Rails em plano.

Fonte: Próprio autor.

Na figura 19 aparece o mesmo sistema criado em Rails. Percebe-se claramente a quantidade de tempo maior que o sistema levou para ser montado, levando vários momentos a utilizar mais o sistema, demonstrando uma instabilidade no uso dos recursos do browser, isso é sinal que além do servidor estar demorando para enviar toda a requisição, o sistema não está bem arquitetado para lidar com o Javascript do browser, pois o projeto utiliza bibliotecas Javascript (*JQuery* e *Bootstrap*) diferente do *Meteor*, que utiliza *Bootstrap* mas tem seu próprio sistema de *templating* no cliente, conhecido como *Blaze*.

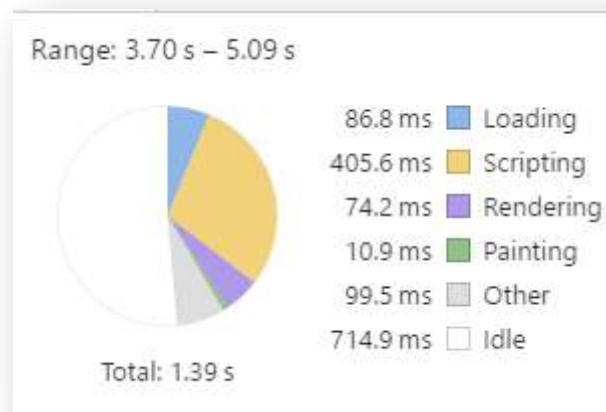


Figure 20: Quantidade de informação inicial em Rails em pizza.

Fonte: Próprio autor.

Na figura 20 é claramente demonstrado o tempo desproporcional que o sistema está sem exercer atividade, isso produz um grande momento em que o usuário se depara com uma tela branca, podendo demonstrar lentidão mesmo com um servidor de testes vazio disponível.

O sistema leva ao todo 5 segundos para ficar pronto, relativo ao 1.5 segundo do Meteor. Diante dessas demonstrações pode-se perceber uma clara vantagem no uso do Meteor, mas não se deve achar necessário usar a plataforma, pois a maior parte dela está no protocolo de comunicação, DDP, ao qual faz todo o balanceamento necessário para que qualquer sistema consiga ser rápido, eficiente e resiliente.

6.5 QUADRO COMPARATIVO GERAL

Os capítulos anteriores demonstraram detalhadamente cada aspecto de comunicação entre clientes e servidores, possibilitando a realização de um quadro

comparativo a fim de visualizar sucintamente os resultados apresentados, veremos como o DDP e o HTTP se comparam em números lado a lado na figura 21.

	HTTP	DDP
Sessão	Simulada (Cookies)	Real
Uso de cabeçalho	Sim	Não
Troca de Informações	Pacotes	Mensagens
Ciclo de vida	Cíclico	Linear
Broadcasting	Push-based	Pull-based
Comando e Busca	Unificada com Pacotes	Separada, um canal para comandar, outro para assinar/receber

Figure 21: Comparativo entre HTTP e DDP.
Fonte: Próprio Autor

7 CONCLUSÃO

O Protocolo de Aplicação DDP, utilizado no MeteorJS provê a ele toda a funcionalidade e rapidez necessária para torná-lo uma opção eficiente e atrativa para o mercado. Diante de tantas soluções prontas que o mercado lança, poucas conseguiram implementar um protocolo em tão baixo nível e que se adapta perfeitamente ao modo de trabalhar do programador Javascript.

Foram vistas três tecnologias contemporâneas do Meteor que tem suas próprias conquistas, o *Java Server Faces*, MEAN e o Rails. As três tecnologias foram mostradas com vantagens que trouxeram na época, e o motivo de sua migração. É perceptível em números a magnitude da mudança na abordagem da aplicação, enquanto a aplicação em Rails obtinha mais de 100 arquivos, a do Meteor menos de 50, é crucial demonstrar que mudanças são constantes na tecnologia, mas mudanças conceituais são raras e alteram todo o aspecto do cenário tecnológico.

Fica nesse relatório o registro da eficácia do DDP, o qual é uma das peças que constroem o Meteor, tendo ainda várias outras que merecem abordagem mais detalhada, pois os seus conceitos também são tão interessantes quanto. A maneira reativa de programar torna o programador mais capaz de desenvolver sistemas que eram inicialmente considerados complexos, mas que na verdade a complexidade foi construída em cima das abstrações que todo framework junto com a indústria entrega, ironicamente torna um caminho de duas rotas, que pode gerar mais esclarecimento ou levar a mais complexidade, mas segue com a mesma intenção de ser o mais simples, e não fácil. É também objetivo desse documento destacar as características diferenciais do protocolo DDP, que é focar na serialização das mudanças no envio de mensagens para o servidor e distribuição das atualizações distributiva e paralelamente. Com essa ideia o foco da pesquisa passa a ser mais abstrato lidando mais na maneira como a informação é distribuída, servindo de idéia para próximas pesquisas que possam lidar com essa área, como *event sourcing* (aplicações sem necessidade de *cache*), ou outras formas de programação.

8 REFERÊNCIAS

A. Melnikov, **RFC 6455 - The WebSocket Protocol**. Disponível em: <<https://tools.ietf.org/html/rfc6455#section-1.1>>

ÄKKIJYRKÄ, Yrkkö; **Dynamic Web-Applications with Meteor.js** - Bachelor of Engineering (Information Technology) Helsinki Metropolia University of Applied Sciences, 2015. Disponível em: <<https://publications.theseus.fi/bitstream/handle/10024/102032/Dynamic%20web%20a%20pplications%20with%20meteor%20Yrkkö%20Akkijyrkä.pdf?sequence=1>> Acesso em: 18/11/2016.

B. Peter, **Matt Debergalis on Meteor**, disponível em <<https://www.infoq.com/interviews/debergalis-meteor>> Acesso em: 23/12/2016

Bossable, **MEAN Stack vs Meteor**. Disponível em: <<http://www.bossable.com/2029/what-is-meteorjs-vs-meanstack/>>. Acesso em 30 de ago. de 2016.

Conceito.de, Robustez, Disponível em: <<http://conceito.de/robustez>> Acesso em: 18/11/2016.

Fowler M., **CQRS**. Disponível em: <<http://martinfowler.com/bliki/CQRS.html>> acesso 14 de outubro de 2016.

Friends A. **About**. <<https://www.apachefriends.org/about.html>>. Acesso em 05/12/2016

Hickey R. **Language of the System**, Disponível em: <https://www.youtube.com/watch?v=ROor6_NGIWU>, acesso 03 de nov. de 2016

HTML5 WebSocket. Disponível em: <<https://dzone.com/refcardz/html5-websocket>> acesso em 13 de outubro de 2016.

Karpov V. **The Mean Stack: MongoDB, ExpressJS, AngularJS and Node.js** <<http://blog.mongodb.org/post/49262866911/the-mean-stack-mongodb-expressjs-angularjs-and>>. Acesso em 05/12/2016.

Kay A., **Alan Kay On Messaging**. Disponível em: <<http://wiki.c2.com/?AlanKayOnMessaging>> acesso em 03 de nov. de 2016.

Kesteren A. V., Aubourg J., Steen H. R. M., **XMLHttpRequest**. Disponível em: <<https://dvcs.w3.org/hg/xhr/raw-file/tip/Overview.html>> Acesso: 28 de set. de 2016

Kleppmann M., **Turning the database inside-out with Apache Samza**. Disponível em: <<https://www.confluent.io/blog/turning-the-database-inside-out-with-apache-samza/>> acesso 07 de novembro de 2016.

Michaelis On-line. Disponível em: <<http://michaelis.uol.com.br/>> acesso em 28 de setembro de 2016

Microsoft Dev Center. **TLS Handshake Protocol**, Disponível em: <[https://msdn.microsoft.com/en-us/library/windows/desktop/aa380513\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa380513(v=vs.85).aspx)>. Acesso em: 19/11/2016

Moir M., **JSF 2.0 Programming Cookbook**. Disponível em: <<http://enos.itcollege.ee/~jpoial/allalaadimised/lugemist/JSF-2.0-Programming-Cookbook.pdf>>. Acesso em 30 de ago. de 2016

N. Strickera *, G. Lanzaa, **The concept of robustness in production systems and its correlation to disturbances.** <<http://www.sciencedirect.com/science/article/pii/S2212827114006507>> Acesso em 05/12/2016

Sashko Stubailo S., **DDP Specification**, Disponível em: <<https://github.com/meteor/meteor/blob/devel/packages/ddp/DDP.md>>
Tanenbaum, Andrew S., **Sistemas Operacionais Modernos**. São Paulo, SP: Pretince Hall, 2009.

Tilkov S. **A Brief Introduction to REST**, Disponível em: <<http://espinosa-oviedo.com/web-programming/wp-content/uploads/sites/5/2014/02/A-Brief-Introduction-to-REST.pdf>>. Acesso em: 19/11/2016

Warbuton R. **Object-Oriented vs. Functional Programming**. Disponível em: <<http://www.oreilly.com/programming/free/object-oriented-vs-functional-programming.csp>>. Acesso em 30 de ago. de 2016

CONTROLE DE COMPROVANTES: Sistema Web Desenvolvido com o Framework JSF

Victor Amaral Freitas
Sheyla Natália de Medeiros

1 INTRODUÇÃO

Existem várias necessidades organizacionais, utilizar um sistema de controle interno é uma das maneiras de manter a empresa em constante crescimento. Apesar da grande necessidade de um controle nos processos, verifica-se que muitas empresas ainda não utilizam-se destes recursos de forma adequada.

No Brasil, temos como modelo controles fiscais que segundo o SEBRAE (Serviço Brasileiro de Apoio às Micro e Pequenas Empresas) “teve sua implementação gradual, iniciada em 2010, tornou-se obrigatória para a maioria das empresas [...]. O objetivo da Receita Federal é a modernização do procedimento, a diminuição de custos e o controle nos processos fiscais” (SEBRAE, 2017).

A emissão de Nota Fiscal eletrônica foi um marco para a padronização eletrônica e o uso de tecnologias entre empresas, além das próprias soluções internas que foram resolvidas com softwares completos, várias funções, diversas melhorias nos setores internos como financeiro, contabilidade, comercial, estoque e na aquisição de serviços de transporte.

Segundo a Agência Nacional de Vigilância Sanitária (ANVISA, 2017) “para se realizar o transporte de medicamentos, a empresa deve obter Autorização de Funcionamento de Empresa (AFE) junto à ANVISA. Caso entre os medicamentos transportados existam medicamentos que contenham substâncias sujeitas a controle especial, a transportadora, além de obter AFE, deverá também obter Autorização Especial (AE).” Deste modo, torna o transporte mais seguro e conseqüentemente mais burocrático. Contudo, o grande problema encontrado é em relação à comprovação da entrega, não existe um sistema para gerir o armazenamento nem a consulta desses comprovantes.

A logística do transporte muitas vezes é realizada por terceiros, são recolhidas as mercadorias junto com a nota fiscal, válido para o transporte e constatação de compra ou venda em postos fiscais. Além disso, nesse tipo de serviço é cobrado o canhoto para comprovação do recebimento total da mercadoria.

Entretanto, algumas transportadoras não realizam esse processo da maneira correta, o canhoto deve ser assinado com o nome e o RG (Registro Geral) de quem está recendo a mercadoria, porém em alguns casos não são protocolados nem arquivados da maneira correta e conseqüentemente dificulta o controle da inadimplência, utilizam o comprovante de entrega para realizar cobranças de dívidas antigas.

Após análise da problemática, este processo pode ser melhorado através de um simples sistema que utiliza as seguintes ferramentas, Mysql que será utilizado para criação do banco de dados que auxiliará no armazenamento das informações, JSF (*JavaServer Faces*) para criação das ações do sistema e o Bootstrap para criação do layout.

“Um sistema de fácil manutenção é um sistema em que as várias responsabilidades estão separadas e implementadas em locais bem-definidos.” (DÉCIO

e ALEXANDRE, 2010, p. 185). Portanto, a divisão do sistema atenderá a perspectiva da arquitetura MVC (*Model, View, Controller*).

Com a utilização dos *frameworks*, é proposto o desenvolvimento de um sistema para gerenciar os comprovantes de entrega, à medida que forem entregues as mercadorias, serão anexadas às fotos dos canhotos junto com as informações de quem está recebendo as mercadorias em um sistema web. Para isso será necessário o empenho dos motoristas que realizam as entregas e a administração interna da transportadora em desenvolver métodos para aplicar a solução resolvendo o problema e disponibilizando as informações no sistema onde todos os clientes da transportadora terão acesso ao canhoto.

1.1 OBJETIVO GERAL

Desenvolver um sistema web, para controlar comprovantes de entrega de mercadoria, utilizando o *framework* JSF.

1.2 OBJETIVOS ESPECÍFICOS

- Levantar requisitos funcionais para criação de diagramas.
- Aplicar arquitetura MVC (*Model-View-Controller*) isolando as regras de negócios da lógica de apresentação.
- Integrar os padrões de projeto DAO (*Data Access Object*), Factory e MVC diminuindo a dependência entre partes do sistema.
- Desenvolver cadastros com *framework* JSF integrado com bootstrap.

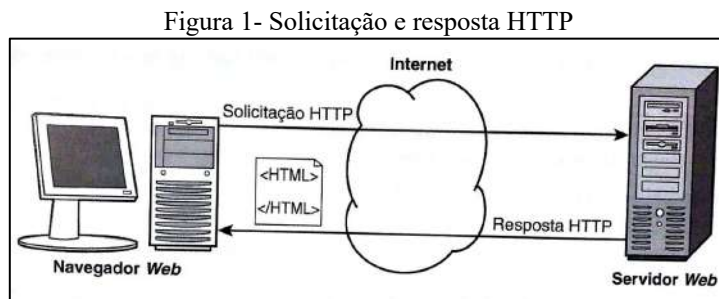
1.3 METODOLOGIA

O trabalho será de natureza prática, descritiva e bibliográfica. Prática por ter a finalidade de produzir um sistema web como produto, descritiva por elaborar, registrar e produzir, bibliográfica por ter como base livros, artigos e revistas. Consisti em uma produção detalhada de um sistema baseado uma arquitetura MVC e os relatos ao decorrer do desenvolvimento com ênfase em sistemas existentes usando *JavaServer Faces*.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 APLICAÇÕES WEB

A Web é uma aplicação cliente/servidor em grande escala, onde o cliente (um navegador Web ou um programa FTP (*File Transfer Protocol*)) se conecta ao servidor usando um protocolo. O mais comum desses protocolos é o HTTP (*Hypertext Transfer Protocol*), onde em uma requisição do *browser* é devolvido pelo servidor textos e imagens (GONÇALVES, 2007). O processo de solicitações e respostas pode ser visualizado pela Figura 1.



Fonte: QIAN et al, 2010, p.3.

HTTP (Protocolo de Transferência de Hipertexto) é um protocolo sem estado que dá suporte a solicitações seguidas de repostas (padrão de troca de mensagens solicitação/resposta)[...] mensagens HTTP também são comumente trocadas entre servidores web ou outros aplicativos que não requeiram interação humana, como um navegador web (QIAN et al, 2010).

As aplicações webs funcionam através das requisições HTTP, conforme apresentado na figura 1, o *browser* envia solicitações HTTP para o servidor responsável pela verificação e responde o que foi requisitado pelo cliente.

2.2 JAVA

A linguagem Java surgiu em 1991 na Sun Microsystems. Inicialmente era parte de outro projeto, chamado Green Project, que tinha como objetivo possibilitar a convergência entre computador, equipamentos eletrônicos e eletrodomésticos. (DÉCIO e ALEXANDRE, 2010).

Segundo Gonçalves (2007) aconteceram evoluções com as mudanças ocorridas no mundo, diversas implementações foram criadas, com intuito de abranger essas mudanças. Hoje é possível utilizar aplicativos *desktop*, páginas para a internet ou até mesmo aplicativos pequenos para celulares, todos desenvolvidos com a linguagem Java.

As principais características do Java são:

- Utilização do paradigma orientado a objetos
- Proximidade com a linguagem humana
- Fortemente tipada
- Estruturada, segura, dinâmica e robusta.
- Variedades de bibliotecas para facilitar o desenvolvimento de sistemas

A existência de *frameworks* Java garante o desenvolvimento de maneira produtiva e possibilitam a junção das principais características da linguagem Java, como o *JavaServer Faces*.

2.3 JAVASERVER FACES

O *JavaServer Faces* (JSF) é a especificação para um *framework* de componentes para desenvolvimento web em Java. Essa especificação foi definida por meio da *Java Community Process* (JCP), que é a entidade que tem como objetivo especificar a evolução da linguagem Java de acordo com o interesse do mercado e não apenas da Sun, criadora da linguagem Java. (DÉCIO e ALEXANDRE, 2010).

JSF é o *framework* Java mais utilizado para o desenvolvimento de aplicações web. Alguns dos principais motivos para isso são: oferece um excelente conjunto de funcionalidades, possui bons materiais de estudo e exige pouco conhecimento inicial para construção de interfaces de usuários tradicionais (FRANCO, 2011).

Para desenvolver usando JSF existem quatro passos básicos:

- Criar classe *Backing Bean*.
- Mapear a classe *Bean*.
- Criar página JSF.
- Realizar o mapeamento entre páginas.

Alguns aspectos melhoram ao utilizar o *framework JavaServer Faces*, maximiza a produtividade, minimiza a complexidade de manutenção e proporciona melhor integração com outras tecnologias web.

2.3.1 Classe *Backing Bean*

A classe Bean (conhecida também como *Backing Bean*) é uma classe Java normal, que suportará todo o funcionamento das páginas JSF. Isso permite que você desenvolva seu projeto separando as regras de funcionamento da organização visual da página e do layout (DÉCIO e ALEXANDRE, 2010).

A classe Bean fornecerá as informações que serão exibidas na página e as operações que serão executadas, mas para que isso ocorra é necessário realizar o mapeamento no arquivo *face-config.xml* ou usando *Annotations*. O exemplo do mapeamento no *faces-config.xml* pode ser visualizado pela figura 2.

Figura 2- Elementos *managed-bean*

```
<managed-bean>
  <managed-bean-name>
    MeuBean
  </managed-bean-name>
  <managed-bean-class>
    meupacote.MeuBean
  </managed-bean-class>
  <managed-bean-scope>
    session
  </managed-bean-scope>
</managed-bean>
```

Fonte: GONÇALVES, 2007, p.451.

O mapeamento visualizado na figura 2 é realizado usando o elemento *managed-bean-name* onde é atribuído um nome, *managed-bean-class* faz referência ao nome da classe instanciada e *managed-bean-scope* define o tempo de vida da instância da classe criada.

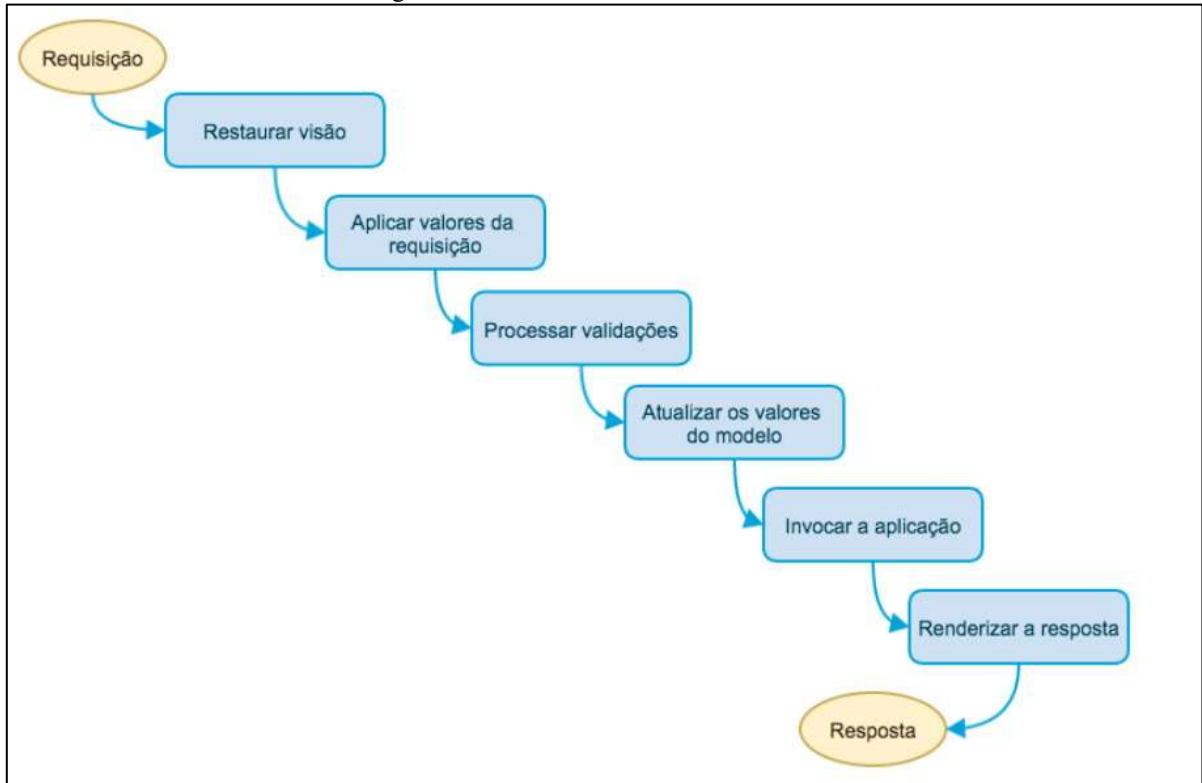
O mapeamento via *annotations* dispensa o uso do arquivo *faces-config.xml*, sendo feito totalmente com declarações na própria classe Java. No lugar do elemento *managed-bean-name* do arquivo xml, utilizará a *annotation* *@managedBean* e no lugar do elemento *managed-bean-scope*, utilizará a *annotation* *@RequestScoped*.

2.3.2 Páginas *JavaServer Faces*

A construção de páginas com o JSF segue uma diferença notável com outros *frameworks* para aplicações em web, o JSF é baseado em componentes, ou seja, ele mantém uma estrutura por trás da tela, e apresenta apenas o código HTML, porém o desenvolvedor não precisa se preocupar a conversão de uma página JSF, o XHTML, para HTML fica a cargo do *framework* (FREIRE, 2014).

Segundo Gonçalves 2007, cada componente pode ser associado com os métodos e atributos de um bean, cada componente também podem ser associados com uma função de validação ou classe, sendo assim um ciclo de vida do JSF é composto por várias fases visualizadas na figura 3.

Figura 3- 6 fases do ciclo de vida JSF



Fonte: FARIA, 2015, p.68.

Conforme Gonçalves (2008) o ciclo de vida é composto:

1 Restaurar a apresentação' — Essa fase inicia o processamento da requisição do ciclo de vida por meio da construção da árvore de componentes do JSF. Cada árvore de componentes possui um identificador único durante todo o aplicativo. O JSF constrói a apresentação da página e salva na instância *Faces-Context* para processamento das fases seguintes.

2 Aplicar os valores de requisição — Nessa fase, quaisquer novos valores inseridos são extraídos e armazenados por seus apropriados componentes. Se o valor do componente não for uma *string*, então ele é convertido para o seu determinado tipo. Se a conversão falhar, ocorrem diversas situações:

1. Uma mensagem de erro é gerada e associada com o componente;
2. Uma mensagem de erro é armazenada no *FacesContext* e depois mostrada posteriormente na fase de Renderizar a Resposta;
3. O ciclo de vida pula a fase de Renderizar a Resposta quando esta se completou.

3 Processar validações — Depois do valor de cada componente ser atualizado, na fase de processo de validações, os componentes serão validados, se necessário. Um componente que necessita de validação deve fornecer implementação da lógica de validação. Por exemplo, em um carrinho de compras, você pode determinar a quantidade mínima a ser digitada e máxima. O valor requisitado é um inteiro, e como passou pela fase 2, nessa fase pode ser barrado por estar além do determinado.

4 Atualizar valores do Modelo — Alcança-se essa fase após todos os componentes serem validados. Nessa fase são atualizados os dados do modelo do aplicativo. Por exemplo, na página que criou para enviar o nome, o que você digitou foi armazenado no bean *MeuBean* durante esta fase. Por ter passado pelo processo de

validação, o valor armazenado será garantido nessa fase. Entretanto, os dados podem violar a lógica de negócios, ao qual a validação ocorre na fase seguinte.

5 Invocar a aplicação — Durante esta fase, a implementação JSF manipula quaisquer eventos do aplicativo, tal como enviar um formulário ou ir a outra página através de um link. Esses eventos são ações que retornam geralmente uma string que está associada a navegação, criada pelo arquivo *faces-config.xml*, no qual se encarrega de chamar a página determinada. É o que ocorreu quando você digitou o nome correto e a página seguinte foi exibida.

6 Renderizar a resposta — Essa é a fase final, ao qual é renderizada a página. Se esse é um pedido inicial para esta página, os componentes são acrescentados à apresentação neste momento. Se esse é um postback, os componentes já foram acrescentados à apresentação. Se há mensagens de conversão ou erros de validação e a página contém um ou mais componentes `<message/>` ou um componente `<messages/>`, esses serão exibidos. Reciprocamente, se a página não contém um componente de mensagem, nenhuma informação aparecerá.

2.4 PADRÕES DE PROJETO

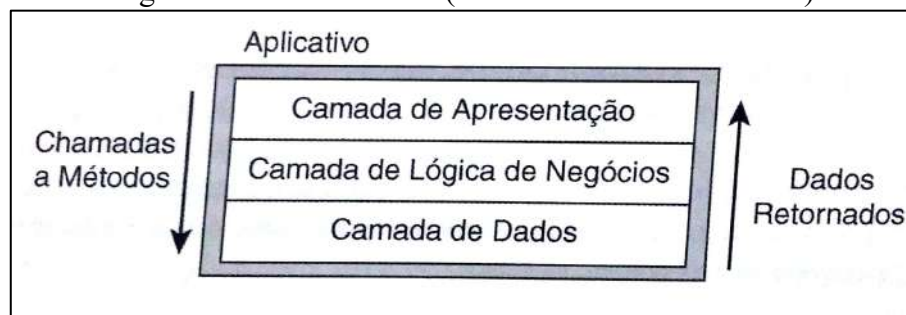
Padrões de projeto são soluções para problemas recorrentes no desenvolvimento de software e que podem ser reusadas por outros desenvolvedores (FRANCO, 2011). Desta forma, os padrões melhoram o acoplamento e a coesão do projeto de software.

2.4.1 Arquitetura MVC

O MVC (*Model – View – Controller*) é um padrão de arquitetura muito utilizado no desenvolvimento de projeto web, mas pode ser aplicado também a qualquer outro tipo de projeto de software que exige integração com usuário (DÉCIO e ALEXANDRE, 2010).

O *JavaServer Faces* (JSF) é um *framework* para desenvolvimento de aplicações web, é baseado no padrão MVC (*Model-View-Controller*), dividindo em três camadas, separando a lógica de negócio, a apresentação e a persistência dos dados (SILVA et al, 2012). Na figura 4 mostra a representação do funcionamento do MVC.

Figura 4- Camadas MVC (Model – View – Controller)



Fonte: QIAN et al, 2010, p.4.

A idéia central do MVC é dividir o programa em três conjuntos, cada um com sua respectiva responsabilidade. Assim, a camada de Visão seria a responsável pela apresentação dos dados. O Controle seria incumbido de receber os dados inseridos pelo usuário, manipulá-los utilizando-se da camada de Modelo, e retornar alguma informação para a camada de Visão. Já o Modelo teria a função de definir o modo como os dados são organizados no meio persistente (MARAFON, 2006).

O padrão MVC possibilita a existência de várias interfaces com o usuário que permitem ser modificadas sem que haja a necessidade da alteração das regras de negócios, possibilitando uma maior flexibilidade.

2.4.2 Data Access Object (DAO)

Segundo Gonçalves (2007) o padrão de projeto DAO (*Data Access Object*) é o padrão mais utilizado para acesso a dados contidos em um banco de dados. O padrão DAO fornece uma interface independente, no qual pode ser usado para persistir objetos de dados.

O componente de negócio que depende do DAO usa a interface mais simples exposta pelo DAO para seus clientes. O DAO oculta completamente os detalhes de implementação da persistência dos dados de seus clientes. Porque a interface exposta pelo DAO para clientes não se altera quando a implementação do código é modificada, este padrão permite que o DAO se adapte a diferentes regimes de armazenamento, sem afetar seus clientes ou os componentes de negócio (OLIVEIRA, 2009). Essencialmente, o DAO funciona como um adaptador entre o componente e a fonte de dados.

2.4.3 Padrão Factory

O padrão *factory* é utilizado para gerenciar a construção dos objetos de uma classe, responsável por determinar qual a classe derivada ou qual implementação de uma interface deve ser instanciada, de acordo com a necessidade de cada cliente (ANDRADE, 2006).

Dessa forma, a *Factory* pode obter uma conexão de dados usando uma fábrica que gerencia as conexões.

2.5 HIBERNATE

O *Hibernate* é um projeto que tem como finalidade ter uma completa solução para o problema de gerenciamento de dados persistidos em Java. Além de ser um *framework* que serve como mediador no relacionamento com banco de dados, relacionamento conhecido como mapeamento objeto/relacional (ORM) para a linguagem Java (GONÇALVES, 2008). A figura 5 demonstra duas técnicas de mapeamento através do XML ou fazendo uso de *Annotations*.

Figura 5-Arquitetura simplificada do *Hibernate*



Fonte: DÉCIO e ALEXANDRE, 2010, p.123.

Hibernate oferece várias ferramentas que ajudam a persistir de forma automatizada e transparente entre elas o ORM (Mapeamento Objeto-Relacional).

2.5.1 Mapeamento Objeto- Relacional

ORM (Mapeamento Objeto-Relacional) é responsável por fazer a transformação de dados de uma forma para outra de maneira reversível. Contudo, utilizando essas decisões sobre a arquitetura de um sistema, técnica de mapeamento tem como consequência uma possível queda de desempenho. Porém existe mais benefícios que prejuízos (DÉCIO e ALEXANDRE, 2010, p.121).

Décio e Alexandre (2010, p.121) explicam que uma solução ORM contém os seguintes pontos:

- Uma API para realizar as operações CRUD básicas em objetos de classes persistentes.
- Uma Linguagem ou API para especificar consultas que se referem às classes persistentes.
- Facilidade de especificar o metadado de mapeamento.
- Uma técnica para que a implementação ORM interaja com objetos transacionais permitindo executar verificações do tipo leitura suja ou carregamento sobre demanda (DÉCIO e ALEXANDRE, 2010, p.121).

Portanto, o *Hibernate* apresenta todas essas características que o torna uma aplicação ORM.

2.6 MYSQL

MySQL é um banco de dados relacional, desenvolvido para plataformas Linux *like*, OS/2, Windows. Sendo um software de livre distribuição para plataformas não-Windows que o utilizam em um servidor Web. MySQL é um servidor multiusuário, multitarefa, compatível com o padrão SQL (*Structured Query language* Linguagem de Consulta estruturada), linguagem essa amplamente utilizada para manipulação de dados em RDBMS (Banco de dados Relacionais), sendo considerada um ferramenta de manipulação de base de dados de tamanho moderado (ORACLE, 2017).

As principais características que destacam MySQL são: seu suporte as instruções SQL; sua natureza de distribuição gratuita; facilidade de integração com servidor Web e linguagens de programação de desenvolvimento de sites dinâmicos.

2.7 BOOTSTRAP

O Bootstrap é um conjunto de ferramentas de código aberto para desenvolvimento com HTML (*Hypertext Markup Language*), CSS (*Cascating Style Sheets*) e JS (*JavaScript*) onde possibilita projetar ideias de vários estilos ou cria um aplicativo completo com *templates* prontos, sistema de grade responsivo, extensos componentes pré-construídos e *plugins* poderosos criados no jQuery (BOOTSTRAP, 2017).

A utilização de classes disponível no Bootstrap facilita na organização dos aspectos visuais de *tags* de demarcação (HTML), possibilitando uma qualidade visual no *front-end* das aplicações web e conseqüentemente o tempo de criação dos layouts será reduzido.

3 SISTEMA WEB DE COMPROVANTES DE ENTREGA

A preparação para desenvolvimento do sistema foi a partir de diagramas de casos de uso baseados no levantamento de requisitos, estrutura formada pela arquitetura MVC, construção do sistema utilizando o *framework* JSF integrado com bootstrap e o Hibernate para persistência de dados no MySql.

3.1 LEVANTAMENTO DE REQUISITOS FUNCIONAIS

O sistema web composto por um administrador que realiza cadastros e alguns funcionários que realizaram consultas e armazenamentos de canhotos no sistema de acordo com os requisitos funcionais apresentados na figura 6.

Figura 6- Requisitos funcionais

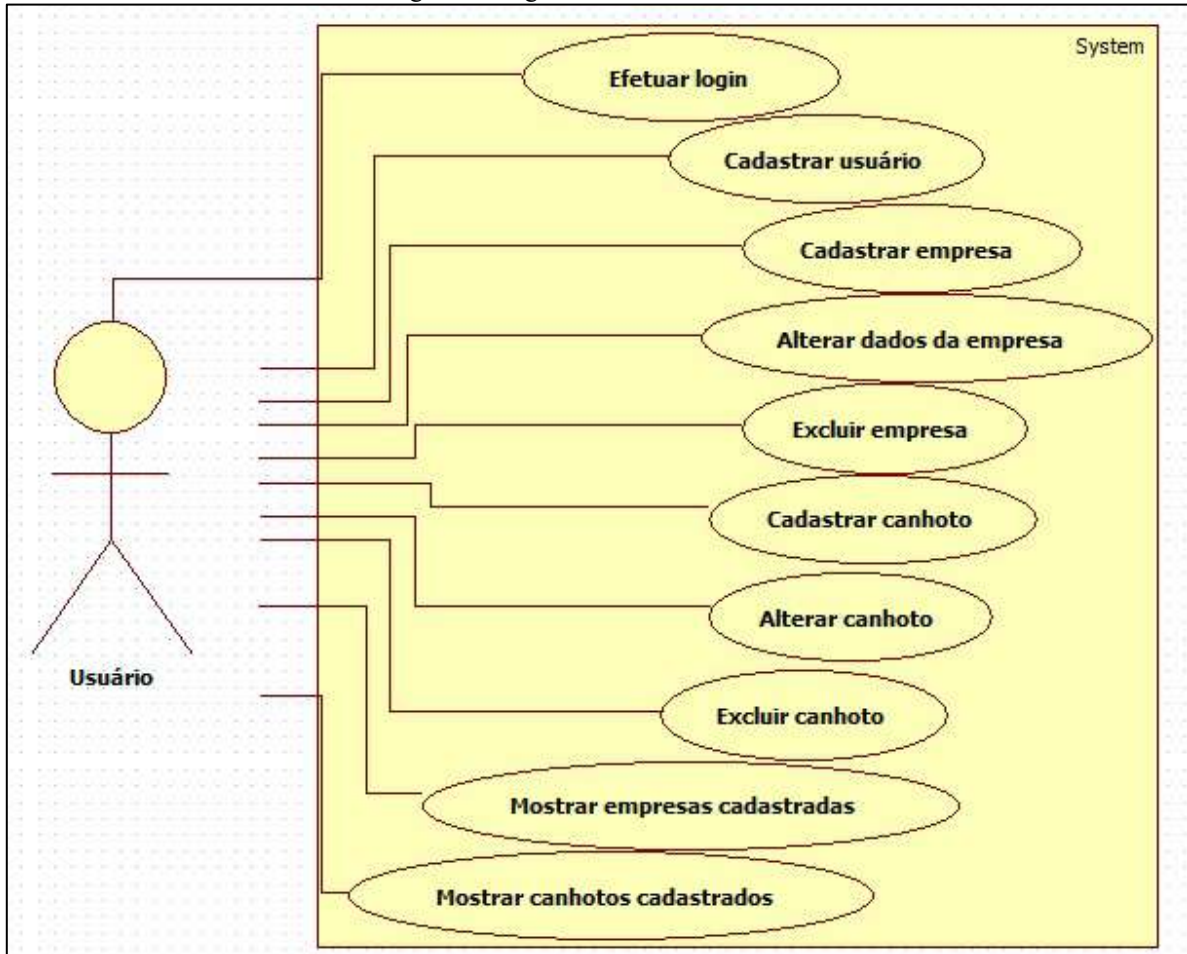
Identificação	Nome	Descrição
RF02	Efetuar login	Efetua login, acesso ao sistema.
RF02	Cadastrar usuário	Um sistema é composto por um funcionário que possui um login e senha para realização dos cadastros.
RF03	Cadastrar empresa	Cadastrar as empresas que terceirizam as entregas de suas mercadorias
RF04	Alterar dados da empresa	Alterar qualquer dado da empresa que já foi cadastrada.
RF05	Excluir empresa	Apagar a empresa por completo.
RF06	Cadastrar canhoto	Cadastrar canhoto de cada nota fiscal que foi entregue com o nome do destinatário que recebeu a mercadoria e o número da nota fiscal.
RF07	Alterar canhoto	Alterar qualquer dado do canhoto que já foi cadastrado.
RF08	Excluir canhoto	Apagar canhoto cadastrado
RF09	Mostrar empresas cadastradas	Mostrar uma lista de todas as empresas cadastradas.
RF10	Mostrar canhotos cadastrados	Mostrar uma lista de todos os canhotos cadastrados.

Fonte: Própria.

3.2 DIAGRAMA DE CASO DE USO

O sistema possui dois atores: o administrador e o funcionário. O Administrador é responsável pelos cadastros de empresa, canhoto e funcionário e pela alteração e exclusão de todos os cadastros. O funcionário acessa o sistema para cadastrar canhotos e consultar canhotos existentes.

Figura 7-Diagrama de caso de uso

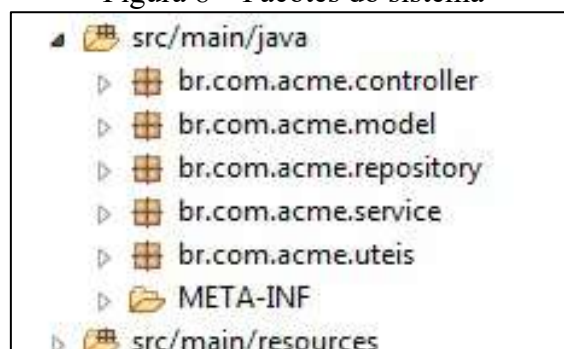


Fonte: Própria.

3.3 IMPLEMENTAÇÃO DO SISTEMA

No início da implementação foram criados pacotes para dividir e implementar os padrões de projeto abordados nos tópicos 2.4.1, 2.4.2, 2.4.3. A divisão dos pacotes caracteriza partes do padrão MVC, conforme a figura 8.

Figura 8 – Pacotes do sistema



Fonte: Própria.

A camada modelo do padrão MVC foi subdividida em dois pacotes: br.com.acme.repository e br.com.acme.service responsável pelo diálogo entre a camada de controle e a própria camada. O pacote br.com.acme.model contém as entidades, cada

entidade possui atributos, conforme apresentado na figura 9, uma das entidades implementadas.

Figura 9 – Entidade usuário

```
@Table(name="tb_usuario")
@Entity
public class UsuarioEntity implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue
    @Column(name="id_usuario")
    private String codigo;

    @Column(name="ds_login")
    private String usuario;

    @Column(name="ds_senha")
    private String senha;

    public String getCodigo() {
        return codigo;
    }
    public void setCodigo(String codigo) {
        this.codigo = codigo;
    }
    public String getUsuario() {
        return usuario;
    }
    public void setUsuario(String usuario) {
        this.usuario = usuario;
    }
    public String getSenha() {
        return senha;
    }
    public void setSenha(String senha) {
        this.senha = senha;
    }
}
```

Fonte: Própria.

Na figura 9 estão presentes as anotações do Hibernate que servem para mapear as classes no banco de dados Mysql e auxiliar a construção, organização e manutenção dos dados. Além do mapeamento cada entidade possui um repositório ou classe DAO que se comunica diretamente com o banco de dados Mysql, conforme figura 10.

Figura 10 – Classe empresaDAO

```

public class EmpresaDAO implements Serializable {

    private static final long serialVersionUID = 1344324234243234L;
    @Inject
    Empresa empresa;
    EntityManager entityManager;
    public boolean gravar(Empresa empresa){
        boolean sucesso = false;
        try {
            entityManager = Uteis.JpaEntityManager();
            entityManager.merge(empresa);
            sucesso = true;
        } catch (Exception e) {
            e.printStackTrace();
        }
        return sucesso;
    }

    public Empresa selecionar(Long codigo){
        Empresa empresa = null;
        try {
            entityManager = Uteis.JpaEntityManager();
            empresa = entityManager.find(Empresa.class, codigo);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return empresa;
    }
    public boolean remover(Empresa empresa){
        boolean sucesso = false;
        try {
            entityManager = Uteis.JpaEntityManager();
            empresa = entityManager.find(Empresa.class, empresa.getCodigo());

            entityManager.remove(empresa);
        }
    }
}

```

Fonte: Própria.

As classes DAO foram implementadas de acordo com o padrão DAO fundamentado no tópico 2.4.2, todas as classes DAO possuem uma instância da fábrica que gerencia as conexões JDBC e suas transações, contém métodos para realização de CRUDs (*Create, Read, Update e Delete*) criação, consulta, atualização e destruição de dados.

A camada de controle está presente no pacote br.com.acme.controller na figura 8, dentro possui as classes que controlam toda as ações solicitadas pela camada de visão e informa para a camada modelo quais os dados devem ser verificados ou persistidos no banco de dados, conforme apresentado na arquitetura MVC. A implementação da classe de controle está contida na figura 11.

Figura 11- Classe empresaBean

```

@Named(value = "empresaBean")
@SessionScoped
public class EmpresaBean implements Serializable {

    private static final long serialVersionUID = 14324242L;

    @Inject
    private EmpresaDAO empresaDAO;
    private transient Empresa empresa;
    private CanhotoDAO canhotoDAO;
    private transient Canhoto canhoto;

    private transient List<Empresa> empresas;

    private transient List<Canhoto> canhotos;

    private Part file;

    public String novo(){
        empresa = new Empresa();
        return "empresa.xhtml";
    }
    @PostConstruct
    protected void init() {
        empresaDAO = new EmpresaDAO();
        empresa = new Empresa();
        canhoto = new Canhoto();
        canhotoDAO = new CanhotoDAO();
        this.canhotos= new ArrayList<>();

    }
    public String gravar() {
        boolean resultado = empresaDAO.gravar(empresa);
        if (resultado) {
            empresa = new Empresa();
            FacesContext.getCurrentInstance().addMessage("", new FacesMessage("Sucesso ,Empresa Cadastrado!"));
        } else {
            FacesContext.getCurrentInstance().addMessage("", new FacesMessage("ERRO!"));
        }
    }
}

```

Fonte: Própria.

Nas classes de controle existem anotações CDI(*Context and Dependency injection*) como `@Named` caracterizado como uma classe *Backing Bean* e `@Inject` que é responsável pela injeção de dependências. As requisições são disparadas pela camada de visão representada pela figura 12.

Figura 12-Formulário de cadastro

```

<h3 class="sub-header">Cadastro de Empresa</h3>
<div class="container-fluid" id="dadosEmpresa">
    <section class="container">
        <div class="container-page">
            <div class="col-md-10">
                <h:form role="form">
                    <messages class="alert alert-success" role="alert" />

                    <div class="form-group col-lg-8">
                        <h:outputLabel value="Razão Social:"/>
                        <h:inputText value="#{empresaBean.empresa.razao}" class="form-control" placeholder="Email address"/>
                        <br/><br>
                    </div>

                    <div class="form-group col-lg-5">
                        <h:outputLabel value="CNPJ:"/>
                        <h:inputText value="#{empresaBean.empresa.cnpj}" class="form-control" placeholder="Idade"/>
                        <br/><br>
                    </div>

                    <h:commandLink action="#{empresaBean.gravar}" class="btn btn-primary pull-right" value="Salvar" update="dadosEmpresa"/>

                </h:form>
            </div>
        </div>
    </section>
</div>

```

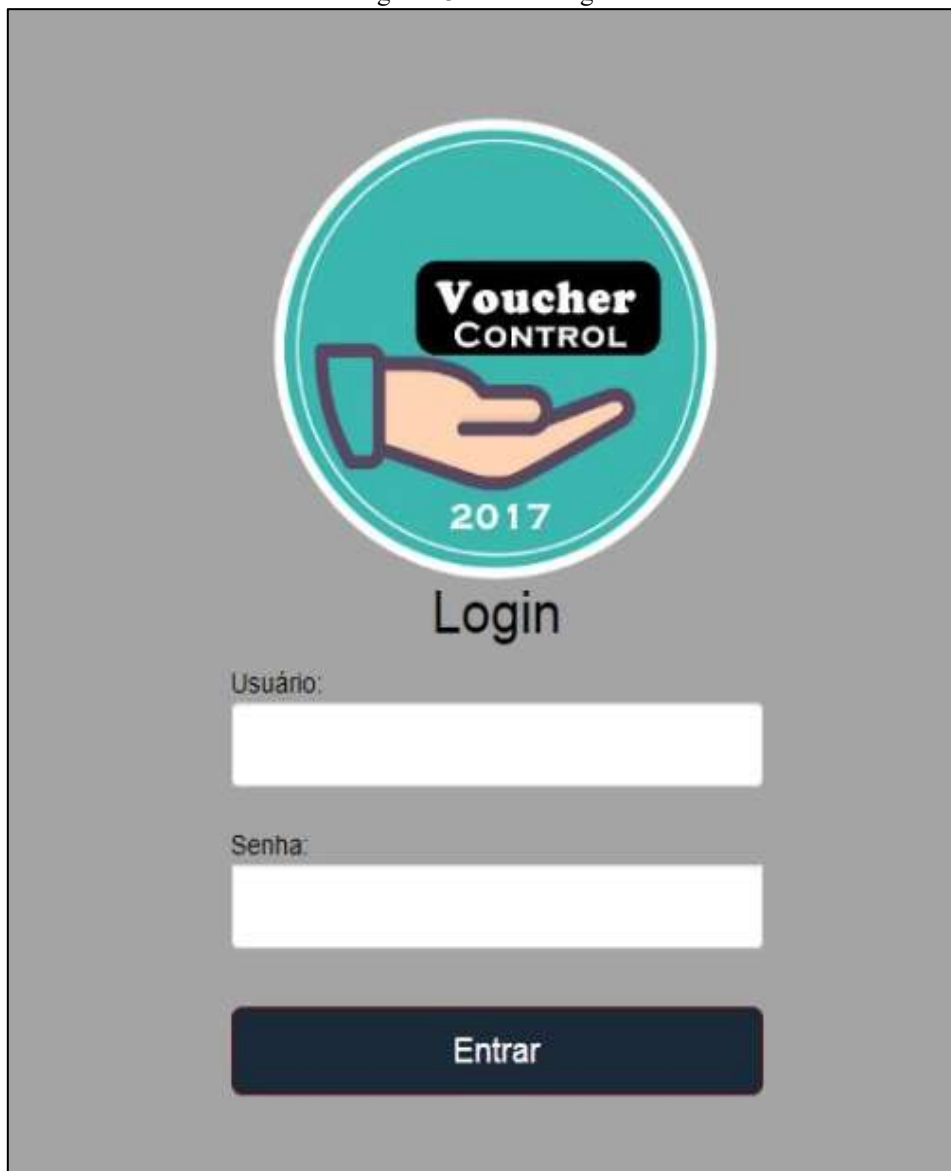
Fonte: Própria.

Na camada de visão conforme a figura 12 estão presentes as classes bootstrap e tags jsf que serão apresentadas no *front-end* da aplicação web.

3.4 APRESENTAÇÃO DO SISTEMA

A tela inicial da aplicação é a tela de login, conforme a figura 13, a partir dessa tela é realizado o acesso ao sistema.

Figura 13- Tela de login



A imagem mostra a tela de login do sistema. No topo, há um logotipo circular com o texto "Voucher CONTROL" e "2017" sobre uma mão estendida. Abaixo do logotipo, o texto "Login" é exibido. O formulário de login contém dois campos de entrada: "Usuário:" e "Senha:". Abaixo dos campos, há um botão "Entrar" em um fundo escuro.

Fonte: Própria.

Para construção das páginas foi utilizado JSF, HTML e *bootstrap*. Ao tentar efetuar o acesso ao sistema, é realizada uma verificação dos dados informados, caso sejam inválidos apresenta uma mensagem de erro, conforme a figura 14.

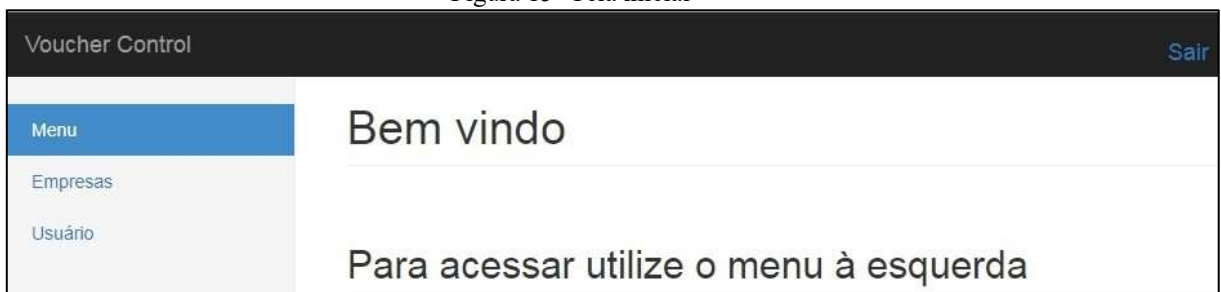
Figura 14 – Login inválido



Fonte: Própria

Ao efetuar o acesso com sucesso (login e senha válidos), o usuário é encaminhado para a tela de apresentação do sistema que é exibida pela Figura 15. O menu na lateral esquerda apresenta as opções (funcionalidades) para o usuário que está logado.

Figura 15- Tela inicial



Fonte: Própria.

O sistema é composto por um menu lateral esquerdo onde estão localizados as opções de cadastros e o setor central que contém o conteúdo da página. Para realização

de um cadastro de usuário é necessário escolher a opção no menu esquerdo, conforme a figura 16.

Figura 16-Página de cadastro de usuário

Voucher Control Sair

Menu
Empresas
Usuário

ACME

Cadastro de Usuário

Nome do usuario:

Senha:

[Salvar](#)

Fonte: Própria.

O cadastro de novos usuários é de responsabilidade do usuário administrador do sistema, cada novo cadastro realizado terá um usuário e senha com acesso total ao sistema. A página de empresas contém a opção de cadastrar novas empresas e cada empresa cadastrada será exibida na mesma página, conforme visto na figura 17,18 e 19.

Figura 17 – Página de empresas

Voucher Control

Menu
Empresas
Usuário

ACME

Empresas

[Novo Cadastro](#)

ID	Razão Social	CNPJ	Inscrição Estadual	Telefone:
----	--------------	------	--------------------	-----------

Fonte: Própria.

Figura 18 – Página de cadastro de empresa

Voucher Control

Menu

Empresas

Usuário

Cadastro de Empresa

Razão Social:

CNPJ:

Inscrição Estadual:

Telefone:

Salvar

Fonte: Própria.

Figura 19 – Página de empresas com uma empresa cadastrada

Voucher Control

Menu

Empresas

Usuário

ACME

Empresas

Novo Cadastro

ID	Razão Social	CNPJ	Inscrição Estadual	Telefone:
4	COMERCIAL E MOURA DE MEDICAMENTOS LTDA - ME	41.226.127/0001-00	16.101.099-7	(83)32923096

Editar Excluir Canhotos

Fonte: Própria.

As empresas que serão cadastradas são aquelas que iram emitir nota fiscal e possuem entrega, já os usuários podem ser funcionários da empresa ou transportadora. Na figura 19 as empresas que estão cadastradas contém 3(três) botões, são eles: editar, excluir e canhotos que serão cadastrados, conforme apresentado pela figura 20.

Figura 20 – Página de canhotos

Fonte: Própria

Na figura 20 o canhoto será cadastrado e anexada uma foto do comprovante de entrega, além de preencher os campos com o nome de quem recebeu a mercadoria e o número da nota fiscal que o canhoto se refere. A foto anexada será salva em uma pasta dentro do sistema para fins de consulta e comprovação que a mercadoria foi entregue. Dessa forma os canhotos podem ser armazenados e visualizados, conforme a figura 21.

Figura 21 – Canhotos cadastrados

ID:	Destinatario:	Recebido por:	RG:	Nota Fiscal:	Empresa Emitente:
1	Victor Amaral Freitas	Joaquim Amaral Freitas	3123129	23213	COMERCIAL E. MOURA DE MEDICAMENTOS LTDA - ME

Fonte: Própria.

4 CONCLUSÃO

O processo de desenvolvimento do *software*, conta com o auxílio de ferramentas como *framework JSF*, *hibernate*, *Mysql* e *bootstrap* que facilitou a criação do sistema web para controle interno dos comprovantes de entrega. A aplicação foi desenvolvida para plataforma web e de grade responsiva que possibilita armazenar os comprovantes de entrega a partir de qualquer dispositivo com acesso a internet.

O *software* possibilita a melhora no processo de busca pelos comprovantes de entrega, descarta a procura e o armazenamento dos canhotos impressos, reduz o tempo de separação e entrega dos canhotos de posse das transportadoras para as empresas que contratam o serviço de transporte.

Durante o desenvolvimento, dificuldades surgiram na integração do framework JSF com o *bootstrap*, são poucos os projetos que trabalham essa integração pelo fato de existir uma biblioteca de componentes, *PrimeFaces* específica para trabalhar com JSF. Contudo, alguns dos problemas como o *upload* de arquivo foi solucionado na versão 2.2 do JSF.

Como trabalhos futuros, pretende-se aprimorar o armazenamento de arquivos, implementar campo de busca avançado que possibilitara a filtragem dos canhotos armazenados, visualização das fotos armazenadas direto no sistema e envio por email da relação das mercadorias entregues.

5 REFERÊNCIAS

ANDRADE, P. M. R. **Sistema de Gerenciamento de Locadora- Sisloc**. Rio de Janeiro, 2006. Disponível em <<http://monografias.poli.ufrj.br/monografias/monopoli10002426.pdf>> Acessado em 07/09/2017.

ANVISA – **Agência Nacional de Vigilância Sanitária**. Disponível em: <<http://portal.anvisa.gov.br/registros-e-autorizacoes/empresas/autorizacao-de-funcionamento/distribuidora-importadora-transportadora>> Acessado em 01/09/2017.
BOOTSTRAP, **bootstrap**. 2017. Disponível em: <getbootstrap.com> Acessado em 10/09/2017.

CARDOSO, G. C. **Logística farmacêutica e o transporte de medicamentos termolábeis**. Rio Grande do Sul, 2016 p.3. Disponível em <<http://revistaseletronicas.pucrs.br/ojs/index.php/graduacao/article/download>> Acessado em 01/09/2017.

FARIA, T. **Java EE 7 com JSF, PrimeFaces e CDI**. 2015 Disponível em <<http://s3.amazonaws.com/algaworks-assets/ebooks/algaworks-ebook-java-ee-7-com-jsf-primefaces-e-cdi-2a-edicao-20150228.pdf>> Acessado em 20/09/2017.

FRANCO, T.S.R. **Estudo comparativo entre frameworks java para desenvolvimento de aplicações web: jsf 2.0, grails e spring web mvc**. Curitiba, 2011. Disponível em <http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/492/1/CT_JAVA_VI_2010_16.PDF> Acessado em 02/09/2017.

FREIRE, S.M. **Ferramenta de apoio a auditoria de programa de segurança de software**. Brasília, 2014. Disponível em <https://fga.unb.br/articles/0000/5474/TCC_Matheus_Freire.pdf> Acessado em 05/09/2017.

GONÇALVES, E. **Desenvolvendo aplicações Web com JSP, Servlets, JavaServer Faces, Hibernate, EJB 3 Persistence e Ajax**. Rio de Janeiro: Ciência Moderna, 2007.
GONÇALVES, E. **Dominando Java Server Faces e Facelets utilizando Spring 2.5, hibernate e JPA**. Rio de Janeiro: Ciência Moderna, 2008.

LUCKOW, D. H; MELO, A. A. **Programação Java para a web: aprenda a desenvolver uma aplicação financeira pessoal com as ferramentas mais modernas da plataforma Java**. São Paulo:Novatec, 2010.

MARAFON, L. D. **Integração javaserver faces e ajax estudo da integração entre as tecnologias jsf e Ajax.** Florianópolis, 2006. Disponível em <https://projetos.inf.ufsc.br/arquivos_projetos/projeto_491/TCC%20-%20Diego%20Luiz%20Marafon.pdf> Acessado em 05/09/2017.

OLIVEIRA, R. R. **Desenvolvimento de um sistema de informação web utilizando padrões de projeto e jsf para um laboratório de análises físico-químicas.** Minas Gerais, 2009. Disponível em <http://repositorio.ufla.br/bitstream/1/5232/1/MONOGRAFIA_Desenvolvimento_de_um_sistema_de_informacao_WEB_utilizando_padroes_deProjeto_e_JSF_para_um_laboratorio_de_analises_fisico-quimicas.pdf> Acessado em 06/09/2017

ORACLE, **ORACLE CORPORATION**, Why MySQL. [s.l. : s.n.], 2011. Disponível em: <http://www.mysql.com/why-mysql>. Acessado em: 10/09/2017.

QIAN, K, ET AL. **Desenvolvimento Web Java.** Rio de Janeiro:Grupogen LTC, 2010.
SEBRAE NACIONAL. **Nota Fiscal eletrônica: tudo o que você precisa saber,** 2017. Disponível em: <<https://www.sebrae.com.br/sites/PortalSebrae/artigos/nota-fiscal-eletronica-tudo-o-que-voce-precisa-saber,b3310d49ac0f3510VgnVCM1000004c00210aRCRD>> Acessado em: 27/08/2017.

SILVA, R. A,GEOFFROY, R, SILVA, N. N. **Sistema web para biblioteca da empresa Marluvas Calçados de Segurança.** MINAS GERAIS, 2012. Disponível em <<http://www.unipac.br/site/bb/tcc/tcc-5e4e08c0a81fbfc69b57b187475a81a2.pdf>> Acessado em 05/09/2017

ESTUDO DE CASO DE BUSINESS INTELLIGENCE: Modelando um Data Mart Sobre Viagens de Transporte Coletivo

Emerson Henrique Soares Silva
Fábio Nicácio de Medeiros

RESUMO

Nos últimos anos, as organizações empresariais vêm percebendo que é preciso pensar em como investir e como se prepararem para serem mais competitivas e eficientes diante dos mercados. Para acompanhar essa nova visão dos empresários os Sistemas de Informações (SI) estão buscando atingir outras necessidades das organizações. Além das necessidades operacionais, buscam satisfazer as necessidades gerenciais e estratégicas dessas empresas. Neste cenário vem ganhando espaço o conceito de *Business Intelligence* (BI), e o objetivo deste trabalho é apresentar conceitos e tecnologias que envolvem o BI e, mais especificamente, apresentar uma fundamentação teórica sobre BI e o que esse termo envolve, como *Data Warehouse* (DW) e suas arquiteturas, *Data Marts* (DM), ferramentas *On Line Analytical Processing* (OLAP) e seus tipos e operações, ferramentas de *Data Mining*, a modelagem de dados dimensional ou multidimensional e as diferenças dessa modelagem em relação à modelagem Entidade-Relacional (ER) dos dados, além de algumas ferramentas para BI disponíveis no mercado. *Data Warehouses* (DW) e *Data Marts* (DM) são tecnologias de armazenamento de dados utilizadas no ambientes de BI, os dados são armazenados sobre tecnologias utilizando, em geral, uma modelagem dimensional. Esses dados são explorados para análise através de ferramentas OLAP e ferramentas de *Data Mining*, que produzem para os usuários executivos, informações que dão apoio ao processo de tomada de decisões. Além de fundamentar esses conceitos, outro objetivo específico é apresentar um estudo de caso de modelagem de um *Data Mart* (DM) sobre viagens de transporte coletivo urbano de maneira explicada e objetiva. Neste estudo de caso é feita uma análise sobre as necessidades levantadas e sobre um modelo ER de um sistema legado. A partir dessa análise é construído de forma elaborada o modelo dimensional de um DM, de modo que sejam observadas técnicas de identificação de fatos, medidas e dimensões, além de técnicas para a elaboração do modelo físico deste *Data Mart*.

Palavras-chaves: *Business Intelligence*. *Data Warehouse*. *Data Mart*. Transporte Coletivo Urbano.

1 INTRODUÇÃO

1.1 Apresentação do problema

Sistemas de transportes coletivos urbanos são responsáveis pelo transporte diário de milhões de pessoas no Brasil, que necessitam desse serviço. Muitas empresas privadas oferecem esses serviços, porém muitas delas não buscam a excelência esperada por seus usuários, por falta de conhecimento não percebem quais são as suas reais necessidades e as formas de oferecer melhores serviços, gastando menos e obtendo mais lucros.

1.2 Justificativa

O uso de tecnologias atuais, unido à visão empreendedora estratégica, pode mostrar as empresas do ramo de transporte coletivo urbano que é possível investir em tecnologias para reduzir custos e ainda oferecer melhores serviços a seus usuários. Para isso é importante conhecer o *Business Intelligence* (BI), e saber quais benefícios as soluções que envolvem o conceito de BI podem trazer de retorno para a empresa.

1.3 Objetivos

1.3.1. Objetivos Gerais

De modo geral, o objetivo desse trabalho é apresentar os conceitos de *Business Intelligence* (BI), e apresentar um estudo de caso sobre a modelagem de um *Data Mart*, que é uma tecnologia de armazenamento utilizada pelo BI.

1.3.2. Objetivos específicos

De modo específico, esse trabalho apresentará os conceitos de BI partindo de suas tecnologias de armazenamento, como *Data Warehouse* (DW), *Data Mart* (DM), e de suas tecnologias de tratamento e apresentação dos dados, como ferramentas *On Line Analytical Processing* (OLAP) e de *Data Mining*.

Em especial, será apresentado nessa produção um estudo de caso sobre construção da modelagem de um *Data Mart* (DM) sobre viagens de transportes coletivos urbanos. Este estudo de caso deve apresentar dicas e técnicas para a construção de modelo conceitual e modelo físico de um DM.

1.4 Metodologia

Pensando em trabalhos futuros, essa produção traz um levantamento teórico sobre os conceitos e tecnologias apresentados nos objetivos, unido a um estudo de caso, que será de utilidade para aprendizado e compreensão da real utilização das tecnologias de armazenamento de dados que envolvem a arquitetura BI, como DW e DM.

O capítulo 2 apresenta a fundamentação teórica sobre os conceitos, tecnologias e ferramentas que envolvem o BI, começando por apresentar definições para BI, seguindo por apresentar conceitos, arquiteturas e características de DW, definições sobre DM, apresentar o modelo dimensional e as diferenças para o modelo ER, apresentar definições sobre OLAP e *Data Mining*, mostrando com poucos detalhes as operações OLAP mais comuns e os tipos de ferramentas OLAP. Nesse mesmo capítulo são apresentadas, também, as fases de um projeto de DW.

No capítulo 3 encontra-se o estudo de caso sobre viagens de transporte coletivo urbano, onde a preocupação é, a partir do entendimento do problema e no estudo de um modelo ER de um sistema legado, identificar fatos, medidas e as dimensões do negócio, a fim de construir elaboradamente, e passo a passo, o modelo conceitual, contendo esses fatos, medidas e dimensões bem relacionadas e representado o modelo dimensional do negócio, para chegar ao modelo físico de implementação de um DM.

No capítulo 4 encontram-se as considerações finais deste trabalho e apresentação das expectativas para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

No cenário atual da Economia da Informação, as organizações empresariais estão se apoiando na tecnologia para buscar novos modelos, novos processos de negócio e novos modos de distribuir o conhecimento, capturando dados, informações e conhecimento que permitam as essas empresas competirem com mais eficiência no mercado. Os Sistemas de Informação (SI) estão atingindo novos espaços nos negócios e na administração, deixando de ser apenas sistemas de apoio operacional para as organizações, atingindo patamares gerenciais e estratégicos, apoiando decisões.

Segundo Barbieri (2001), um conceito ganha espessura no cenário dos negócios, o conceito de *Business Intelligence* (BI), ou Inteligência de Negócios. BI pode ser entendido como um guarda-chuva conceitual que envolve Inteligência Competitiva, ou *Competitive Intelligence* (CI), Gerência de Conhecimentos, ou *knowledge Management System* (KMS), *Internet Business Intelligence* (IBI), pesquisa e análise de mercados, etc.

2.1 Business Intelligence

O conceito de *Business Intelligence* (BI), de modo geral, envolve utilizar dados de diversas fontes dentro de uma organização, de forma a tratar esses dados e obter através deles informações que permitam definir estratégias de competitividade nos negócios.

Barbieri (2001) afirma que BI “representa a habilidade de se estruturar, acessar e explorar informações, normalmente guardadas em *Data Warehouses* e *Data Marts*, com o objetivo de desenvolver percepções, entendimentos, conhecimentos, os quais podem produzir um melhor processo de tomada de decisão”.

BI é um conceito empregado a ferramentas, tecnologias e metodologias, que tem como objetivo fornecer informações estratégicas para suporte a tomada de decisões nos negócios [FELBER, 2005].

Business Intelligence é um termo genérico para descrever o levantamento de informações sobre os ativos internos e externos da organização para tomar melhores decisões de negócios [KIMBALL e ROSS, 2002].

Para Moss e Atre (2003) o BI não é um produto, nem um sistema. É uma arquitetura e uma coleção de aplicações e bancos de dados operacionais integrados, bem como suporte a decisão, que prover a comunidade empresarial fácil acesso aos dados do negócio.

O BI envolve uma infra-estrutura que possui a camada ETL (*Extract Transform Load*), camada essa que provê meios e ferramentas para extrair, transformar e carregar os dados de outras fontes para uma base única, não volátil, organizada por assuntos e que varia com o tempo, uma tecnologia de armazenamento chamada *Data Warehouse* (DW). O DW é construído sobre um modelo diferente do convencional modelo relacional dos sistemas legados e emergentes. Trata-se do modelo dimensional, que é trabalhado sobre tabelas, porém organizadas com uma estrutura central conhecida como tabela de fatos, que é basicamente definida por valores que representam os indicadores estratégicos do negócio, e com estruturas periféricas que são as tabelas de dimensões que se relacionam com a tabela de fatos, as tabelas de dimensões não possuem valores, e sim dados descritivos sobre determinados fatos [KIMBALL, 1998]. A organização dos dados sobre o modelo dimensional permite a interpretação dos mesmos sobre uma visão analítica através de ferramentas OLAP (*On Line Analytical Processing*), que se caracterizam por permitir análises estratégicas dos processos de negócio que possam ser

convertidas em tomadas de decisões, e/ou permite essa interpretação sobre o prisma inferencial através de técnicas de *Data Mining*, que se caracterizam por detecção e análise de padrões para construção do conhecimento.

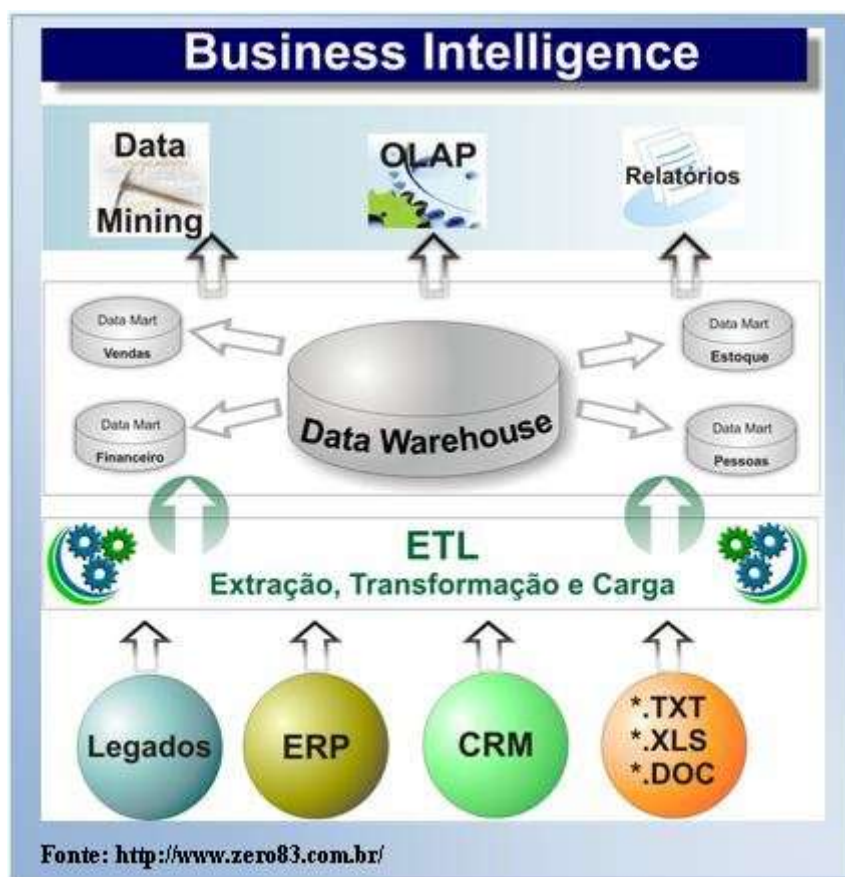


Figura 22. Arquitetura de BI.

A figura 1 apresenta o funcionamento desta arquitetura que é o BI. Em uma organização os dados podem estar espalhados em sistemas legados², nos emergentes sistemas ERP³, nos CRM⁴, em documentos de texto e planilhas. Esses dados são então extraídos de suas fontes, transformados e compilados para um padrão de formato da organização, e depois carregados em *Data Warehouses* e/ou *Data Marts*. Estes repositórios são acessados por ferramentas de relatórios, de processamento analítico (OLAP) e/ou de mineração de dados (*Data Mining*), para então vir a apresentar

² Sistema legado é qualquer sistema de informação que resiste significativamente à modificação e a evolução. Geralmente utilizam bancos de dados obsoletos [BRODIE e STONEBRAKER, 1995].

³ ERP (*Enterprise Resource Planning*) ou SIGE (*Sistemas Integrados de Gestão Empresarial*, no Brasil) é um são sistemas de informação que integram todos os dados e processos de uma organização em um único sistema [LAUDON][PADOVOZE].

⁴ Customer Relationship Management (CRM), ou Gestão de Relacionamento com o Cliente, é uma expressão que defini toda uma classe de ferramentas que automatizam as funções de contato com o cliente, essas ferramentas compreendem sistemas informatizados e fundamentalmente uma mudança de atitude corporativa, que objetiva ajudar as companhias a criar e manter um bom relacionamento com seus clientes armazenando e inter-relacionando de forma inteligente, informações sobre suas atividades e interações com a empresa [ANDERSON e KERR, 2001].

informações e conhecimento que contribuam com a tomada de decisão da organização.

Nas próximas seções trataremos um pouco mais sobre as ferramentas e tecnologias que integram o conceito de BI.

2.2 Data Warehouse

Sobre *Data Warehouse* (DW) ou Armazéns de Dados, Machado (2007) afirma que representa uma grande base de dados capaz de integrar, de forma concisa e confiável, as informações de interesse para a empresa que se encontram espalhadas pelos sistemas operacionais e em fontes externas, para posterior utilização nos sistemas de apoio à decisão.

Segundo Barbieri (2001), *Data Warehouse* pode ser definido como um banco de dados, destinado a sistemas de apoio a decisão e cujos dados foram armazenados em estruturas lógicas dimensionais, possibilitando o seu processamento analítico por ferramentas especiais (OLAP e *Mining*).

O DW integra dados de várias fontes de informações incompatíveis em um repositório de dados único e consolidado, permitindo que sejam retiradas informações que se transformarão em conhecimento após análise precisa e consistente dos administradores da empresa [SING, 2001 *apud* ZORZIN, 2006].

O DW é a representação da camada de armazenamento da arquitetura do BI sobre banco de dados. Para Machado (2007), construir um DW representa uma quebra do paradigma acadêmico sobre construção de banco de dados, já que o DW não se baseia em operações orientadas a transações de negócios. Com isso a integridade referencial deixa de ser a maior preocupação que passa a ser a capacidade de representar o negócio historicamente, a fim de se obter e analisar informações que permitam melhorar e agilizar a capacidade de tomada decisões nos negócios.

O resultado de um projeto de DW, segundo Machado (2007), é:

- informação disponível para gestão;
- visão de curvas de comportamento;
- agilidade de ferramentas de apoio à decisão;
- segurança de informações para decisão;
- maior abrangência de visão de indicadores;
- recursos mais abrangentes para análise de negócios;
- e necessidade e expectativas executivas atendidas por tecnologia da informação.

Machado (2007) justifica a necessidade de construir um DW em uma organização, se nessa existirem várias plataformas de hardwares e softwares, se ocorrem constantes alterações nos sistemas transacionais corporativos, se existe uma dificuldade acentuada na recuperação de dados históricos em períodos anteriores ao ano atual de operações, se existem sistemas de diferentes fornecedores funcionando na organização, se falta padronização e integração entre os dados existentes nos diversos sistemas, se existe uma carência em segurança no armazenamento dos dados, e se existe dificuldade de aplicação de sistemas de informações executivas devido a dependências múltiplas de sistemas corporativos.

2.2.1. Características dos Data Warehouses

Segundo [INMON, 1997 *apud* ZORZIN, 2006], um DW é um conjunto de dados **orientados por assuntos, integrados, não voláteis, variáveis em relação ao tempo** e que **apóiam a tomada de decisões**. As características apresentadas na definição acima representam quais devem ser as preocupações de um projeto de DW. Nas subseções a

seguir serão explicados os significados de cada uma das características apresentadas na definição acima.

2.2.1.1 Dados orientados por assuntos

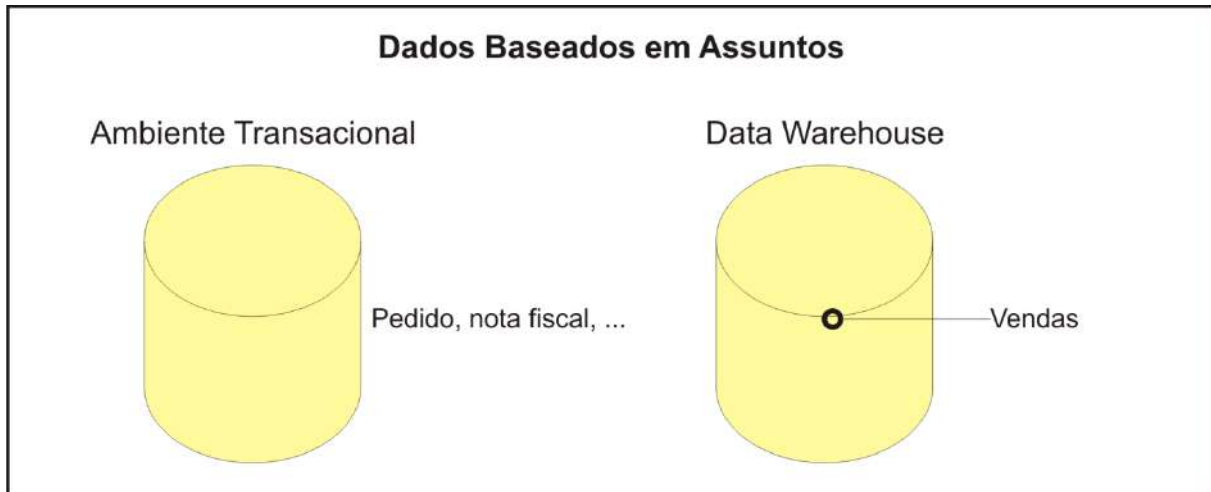


Figura 23 - Orientação por assunto (Baseado em [MACHADO, 2007]).

Os sistemas transacionais operacionais trabalham com os dados do dia-a-dia, das atividades que envolvem o processo de funcionamento da organização, dados que existem basicamente para fins de controle operacional e no geral não apóiam decisões de negócio. Um exemplo são os dados de um sistema de vendas, que se preocupam com realizar pedidos e gerar notas fiscais.

Para dar suporte à tomada decisão, seria preciso, por exemplo, um DW chamado Vendas, onde existiriam dados que pudessem informar o quanto de um produto foi vendido em um determinado período, o quanto de lucro a venda desse produto gerou. Veja na figura 2 que no DW não há preocupação com dados do processo de vendas, como pedidos e notas fiscais, mas sim com o assunto vendas.

2.2.1.2 Dados integrados

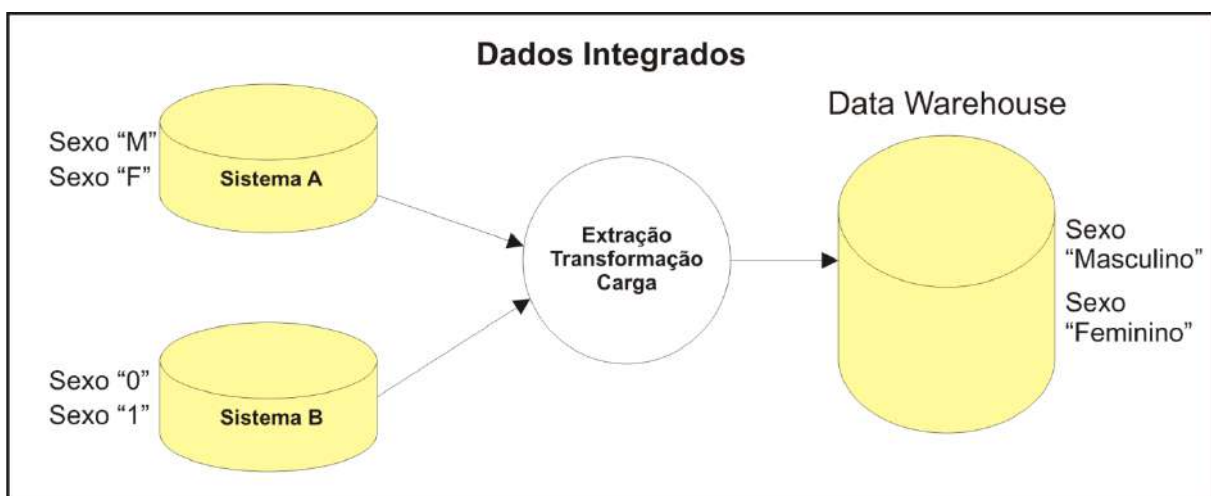


Figura 24 - Integração dos dados (Baseado em [MACHADO, 2007]).

A integração dos dados é uma característica essencial para manter o DW consistente e coeso. O DW é alimentado por outras fontes de dados, que podem trazer dados representados de formas diferentes, mas que possuem o mesmo significado, por exemplo, em uma base de dados **A** o dado sexo está representado pelos caracteres M ou F (M para masculino e F para feminino), mas na base de dados **B** os campos que definem o sexo estão representados por 0 ou 1 (0 para masculinos e 1 para feminino). Não podemos ter um mesmo dado representado de formas diferentes no DW, por tanto no processo de extração e transformação esses dados devem ser unificados e padronizados, afim de não gerar inconsistências no DW. Por tanto, fica definido no exemplo da figura 3, que os dados para um campo sexo serão representados no DW como Masculino e Feminino.

2.2.1.3 Dados não voláteis

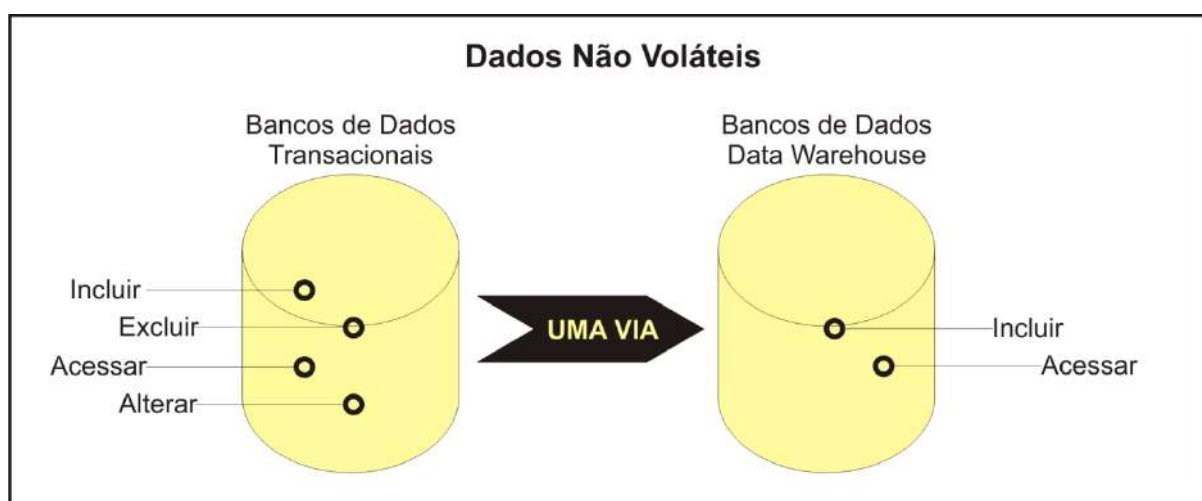


Figura 25 - Dados não voláteis (Baseado em [MACHADO, 2007]).

Em bancos de dados transacionais os dados são trabalhados de maneira extremamente dinâmica, isto é, nos bancos de dados transacionais ocorrem processamentos de inclusão, exclusão, alteração e acesso aos dados frequentemente. O DW só permite duas operações básicas: a inclusão inicial e incremental de novos dados e o acesso somente para leitura a estes dados, isto é, não existem alterações nem exclusões dos dados do DW, isso para evitar impasses e atualizações registro a registro.

2.2.1.4 Dados variáveis em relação ao tempo

O DW deve ser carregado periodicamente através de um processo *batch* (em lotes), e deve ser armazenada uma ou mais referências temporais sobre os dados carregados, de modo que seja possível verificar o histórico desses dados. O tempo é uma característica importante para tomada de decisões, por isso os dados do DW devem variar em relação a suas referências de tempo.

2.2.1.5 Dados que apóiam a tomada de decisões

O DW é construído sobre um modelo multidimensional que se caracteriza por centralizar os fatos pelas dimensões dos negócios. Esse possui a característica de organização lógica dos negócios da organização, o que permite um acesso facilitado de

ferramentas para análise dos negócios, e o resultado dessas análises permitem visualizar cenários passados, atuais e possíveis da organização, gerando informações estratégicas que possibilitem a tomada de decisões de melhorias e evoluções. O modelo multidimensional será mais bem tratado na seção 2.6 deste capítulo.

2.3 Data Mart

Um *Data Mart* (DM), ou Mercado de Dados, é um subconjunto de um DW de empresa inteira. Desempenha um papel de DW departamental, regional ou funcional, apresentando as mesmas características de DW. E uma empresa pode construir uma série de DM ao longo do tempo e vinculá-los a um DW lógico de empresa inteira [SING, 2001 *apud* FELBER, 2005].

Sobre os DMs, Felber (2005) afirma que estes são muito bem aceitos no campo empresarial por exigirem menor investimento em infra-estrutura, produzirem resultados mais rapidamente e por serem escaláveis até DW.

De acordo com Machado (2007), “o DM, normalmente, é modelado em um esquema estrela de acordo com as necessidades do usuário final”. Ainda segundo Machado (2007), empregar um DM possibilita retorno rápido sobre investimento e garante um maior envolvimento do usuário final, que é capaz de avaliar os benefícios extraídos do investimento.

2.4 Arquiteturas e abordagens de implementação para DW

A abordagem de implementação e a arquitetura escolhida certamente provocará impactos diretos no sucesso do projeto, seja positivamente ou negativamente. De acordo com Machado (2007), várias variáveis devem ser analisadas para a escolha da implementação e da arquitetura, dentre elas: o tempo para execução do projeto; o retorno do investimento a ser realizado; a velocidade dos benefícios da utilização das informações; a satisfação do usuário executivo; e os recursos necessários à implementação de uma arquitetura.

De acordo com Barbieri (2001), os primeiros projetos de DW, provavelmente, se guiaram por metodologias apresentadas por dois gurus da década de 1990, Bill Inmon e Ralph Kimball.

2.4.1. Abordagens iniciais

A abordagem de Bill Inmon, de acordo com o apresentado por Machado (2007), era de uma implementação *top-down* (do topo para a base), o que significa a construção inicial de um DW como um grande depósito central de informações empresariais filtradas, limpa e integradas, e de onde outros depósitos secundários departamentais (DM) seriam originados e construídos.

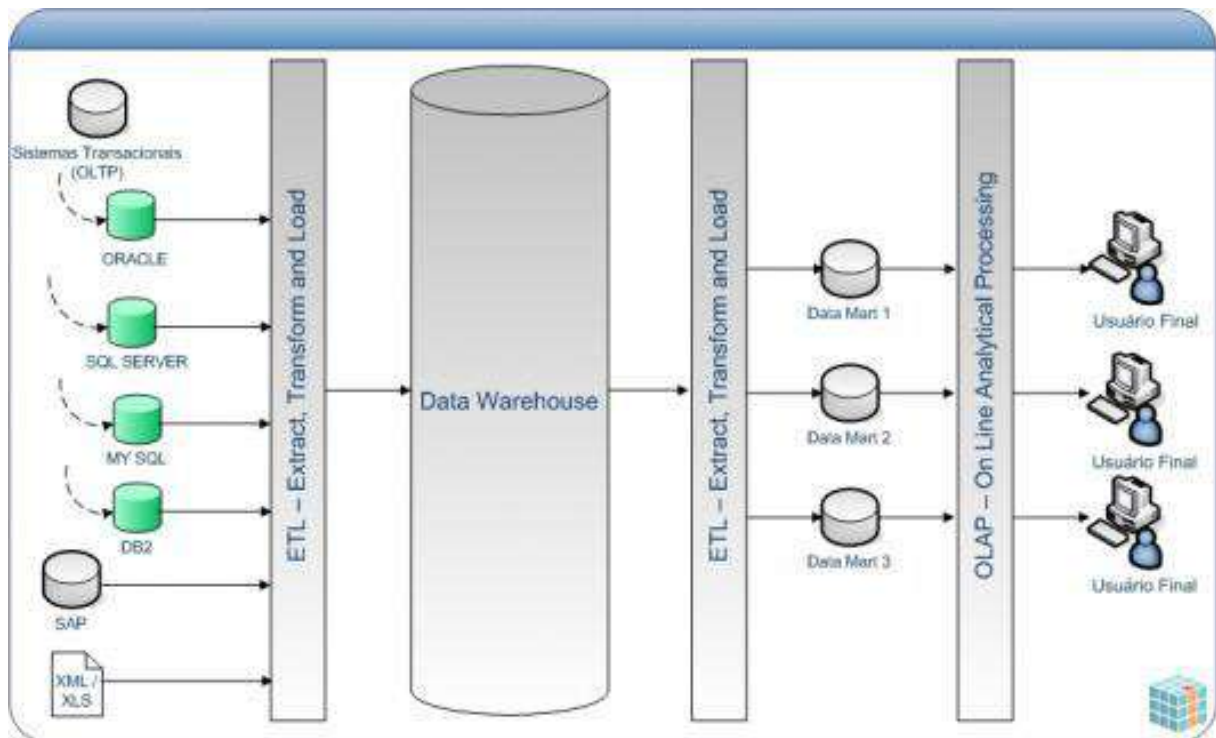


Figura 26 - Implementação top-down [BONOMO, 2009].

A figura 5 apresenta um modelo de implementação com essa abordagem. Bill Inmon apresentou, também, um ciclo metodológico básico que se iniciava pela análise de requisitos de negócio (*BRA – Business Requirements Analysis*), em seguida a modelagem de dados, e posteriormente a análise das fontes de dados, seguindo uma etapa de projeto, a especificação, construção, teste, validação e implantação. Segundo Machado (2007) as vantagens dessa abordagem são: heranças de arquitetura, já que os DM são originados de um DW e estes utilizam a arquitetura e os dados desse DW, o que permite uma facilidade de manutenção; visão de empreendimento, pois o DW concentra toda a empresa e seus negócios; controle e centralização de regras, isto é, um único conjunto de aplicações para extração, transformação e integração dos dados, além de processos centralizados de monitoração e manutenção. Como desvantagem, essa abordagem apresenta uma implementação muito longa, o que significa que demora muito para que esse DW venha a entrar em produção, e a consequência disso é o alto risco de investimento, além da demora do retorno sobre esse investimento.

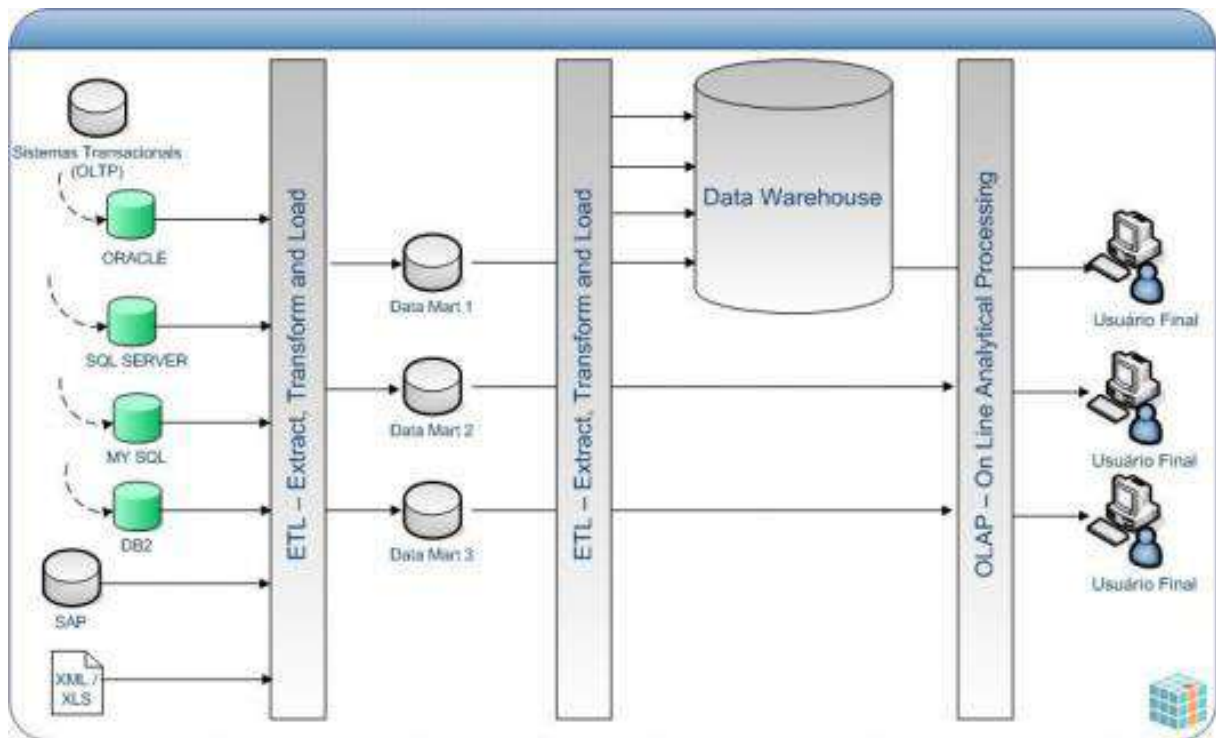


Figura 27 - Implementação na abordagem bottom-up [BONOMO, 2009].

A abordagem de Ralph Kimball é oposta a de Inmon, de acordo com Machado (2007), é uma abordagem *bottom-up* como mostrada na figura 6 acima, que trata do desenvolvimento de projetos de *Data Marts* separados que deverão ser integrados na medida da evolução dos mesmos. Mas a essência da abordagem de Kimball é centrada na etapa de projeto dos *Data Marts*, mais especificamente na modelagem de dados, onde ele apresenta o modelo dimensional *star schema*, do qual vamos tratar na próxima seção. Por ser mais simples e incremental essa abordagem tinha como vantagem uma implementação mais rápida, com um retorno sobre o investimento mais rápido, o enfoque da equipe é de manutenção mais fácil, além de que pode existir uma herança incremental, isto é, um DM pode herdar características de outros já construídos permitindo o reaproveitamento. Porém, como desvantagem, existiria a possibilidade de que os DM produzidos não possuíssem uma perfeita coesão e inviabilidade de integração, além disso, existiria, também, a provável repetição de esforços na fase de extração, transformação e carga dos dados.

Essas abordagens amadurecem sobre a perspectiva histórica dos projetos de DW, o que acabou provocando uma convergência dessas abordagens. O insucesso de projetos sobre a abordagem *top-down* que demoravam mais do que o suportado pela gerência para apresentar informações de suporte às decisões. O insucesso de projetos desenvolvidos sobre a abordagem *bottom-up* que satisfaziam apenas parte da demanda e apresentavam riscos de criação de ilhas de informações, ou seja, apenas parte da organização era beneficiada com informações de suporte a decisão e não existia integração com o resto dos negócios da mesma. O amadurecimento se dá com percepção de que deveria se aproveitar do melhor dos dois mundos, isto é, obter uma visão das informações de toda a empresa e seus negócios de forma integrada, porém construindo de forma incremental e por áreas de negócio. Isto é, inicialmente seria construído o primeiro DM, e gradativamente seriam construídos os demais DM integrados aos anteriores e assim chegariam a um grande repositório integrado e coeso,

um DW. Uma construção incremental e integrada, de modo que as dimensões sejam alinhadas e conformes para todas as áreas, e possuam métricas compatíveis.

2.4.2. Tipos de Arquiteturas

A escolha da arquitetura é uma decisão gerencial do projeto, e está normalmente baseada nos fatores relativos à infra-estrutura disponível, ao ambiente de negócios (parte da empresa), concomitantemente com o escopo de abrangência desejado, assim como a capacitação dos empregados da empresa e dos recursos disponibilizados ou projetados para investimento. (MACHADO, 2007, p.47)

Machado (2007) apresenta em sua obra três arquiteturas para projeto de DW. São elas: arquitetura **global** (centralizada e descentralizada), arquitetura de **Data Marts independentes** e arquitetura de **Data Marts integrados**.

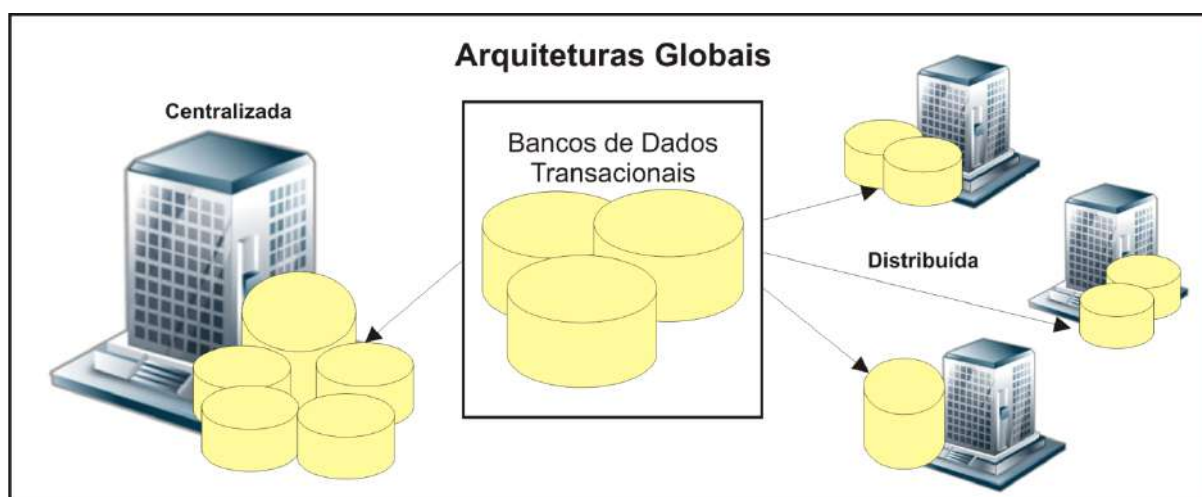


Figura 28 - Arquitetura global. Centralizada e distribuída [MACHADO, 2007].

Na **arquitetura global** o projeto é feito com base nas necessidades da empresa como um todo e o objetivo é construir um repositório de dados de suporte a decisão disponível para toda a organização. Porém isso não significa que esse projeto seja baseado na abordagem *top-down* e nem que estamos falando de um único repositório físico centralizado, o objetivo é disponibilizar o acesso aos dados por toda a organização. A figura 7 apresenta a arquitetura global centralizada e distribuída. Centralizar ou distribuir muito tem a ver com a infra-estrutura física da empresa no aspecto de tecnologia da informação, se ela está instalada em um único local, ou se está distribuída em várias localidades. A grande vantagem desta arquitetura é que os usuários podem utilizar visões corporativas dos dados, porém, como desvantagem, o consumo de tempo de desenvolvimento e administração de um ambiente sobre esta arquitetura é muito grande, assim como o custo de implementação.

Na **arquitetura de DM independentes** não existem conexão entre os DMs, que são isolados por departamento ou por grupos específicos de usuários. Não possuem foco corporativo algum. A interferência do setor de tecnologia da informação da empresa é mínima no que diz respeito a controle de implementação e desenvolvimento do(s) DM(s), visto que esse setor preocupa-se apenas com a manutenção técnica do ambiente.

Esse tipo de arquitetura geralmente resulta em uma implementação bastante rápida devido ao escopo reduzido e isolado, porém fica claro que a capacidade de decisão através de dados corporativos é limitada.

Na **arquitetura de DM integrados** os DMs são distribuídos em departamentos ou áreas de negócios como na arquitetura de DMs independentes, porém os dados não são mais isolados, são interconectados, integrados e acessíveis por outros departamentos ou áreas, por tanto essa arquitetura provê um aumento na capacidade de visão corporativa e maior qualidade das informações do que a anterior, porém aumenta o nível de complexidade dos requisitos, aumenta a necessidade de controle e de administração, e certamente aumenta o envolvimento do setor de tecnologia da informação da empresa.

2.5 Granularidade de Dados

Em projetos de DW, a granularidade de dados é um dos fatores mais importantes a serem tratados, pois afeta profundamente o volume de dados e os tipos de consultas a serem realizadas.

Granularidade, nesse contexto, refere-se ao nível de detalhamento ou resumo disponíveis nos dados. Quanto mais detalhes os dados envolvem, menor é a granularidade desses dados, e quanto mais resumidos, maior é a granularidade.

A razão da análise dessa característica ser tão crucial em um projeto de DW/DM é que quanto menor a granularidade, maior o detalhamento dos dados e maior o volume de dados no DW/DM, isso implica perda de performance nas consultas, porém uma menor granularidade atende um número superior de possibilidade de consultas.

Machado (2007) aconselha estabelecer a granularidade dos dados que deverão compor o DW/DM antes de realizar a modelagem dos mesmos, pois assim o processo de modelagem ocorrerá de forma mais tranqüila e facilitada.

A figura 8 exemplifica o alto e o baixo nível de granularidade dos dados.

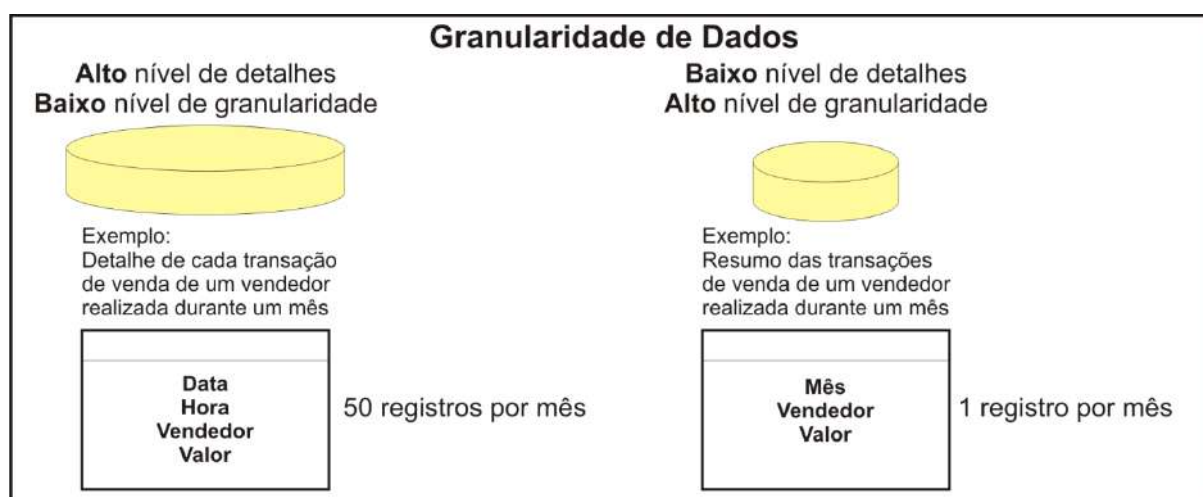


Figura 29 - Granularidade de dados (Baseado em [MACHADO, 2007])

2.6 O modelo dimensional ou multidimensional

As subseções a seguir, apresentam o modelo dimensional tratando os principais conceitos, as características, as diferenças entre este e o modelo ER (Entidade-Relacional), os esquemas propostos sobre essa modelagem.

2.6.1. Conceito

De acordo com Machado (2007) a modelagem multidimensional é uma técnica de concepção e visualização de um modelo de dados de um conjunto de medidas que descrevem aspectos comuns de negócios. E esse modelo é formado por três elementos básicos: os fatos, as dimensões e as medidas (variáveis).

O objetivo dessa modelagem é fornecer a capacidade de visualizar os dados de uma organização, de modo a permitir a análise de valores desses dados, isto é, permitir obter informações de apoio a decisão.

2.6.1.1 Fatos

Os fatos definem a importância e a motivação da modelagem dimensional. Representam numericamente os valores que refletem temporalmente a evolução dos negócios de uma organização. No modelo dimensional um conjunto de fatos estão representados em tabelas denominadas tabelas de fatos [MACHADO, 2007].

Observe a seguinte frase: “O índice de vendas de álcool em gel da marca X vem aumentando no último trimestre”. O índice e vendas é uma relação algébrica da quantidade do produto álcool em gel da marca X vendidas no último mês, sobre quantidade vendida nos três meses anteriores. É uma relação que representa numericamente a quantidade de um produto vendida em um determinado período de tempo. Portanto este índice de vendas representa um fato.

Os fatos são compostos por dados de medidas e de contexto. Segundo Machado (2007), medidas são atributos numéricos que representam um fato, o desempenho de um indicador de negócios relativo às dimensões que participam do fato. Como exemplo, o valor em real das vendas, as unidades vendidas de um produto, o custo em real da venda, entre outros.

2.6.1.2 Dimensões

Dimensões são grandezas de negócios, são elementos pelos quais os fatos são envolvidos, são assuntos de negócios.

De acordo com Machado (2007), as dimensões determinam o contexto de um assunto de negócios. Um exemplo de assunto de negócios é Vendas, e foi apresentado na figura 2. As dimensões que envolvem os fatos das vendas de produtos podem ser:

- Tempo;
- Lojas;
- Clientes;
- Vendedores;
- Seções;
- Produtos;
- Etc.

Segundo Kimball (1998), “uma dimensão típica contém uma ou mais hierarquias naturais, além de outros atributos que não possuem um relacionamento hierárquico com qualquer dos atributos da dimensão”. Hierarquia, nesse contexto, é a classificação dos dados dentro da dimensão. Um exemplo se dá na dimensão tempo, por exemplo: em uma determinada hierarquia, podemos organizar a dimensão nos níveis ano, trimestre, mês, dia; em outra hierarquia poderíamos organizar a dimensão nos níveis semana e dia.

Essa organização se apresenta de forma interessante, visto que uma mesma semana pode pertencer a dois meses, dois trimestres e até dois anos [MACHADO, 2007].

A dimensão é, também, representada em tabela, tabela periférica que descreve uma característica de um fato, quando, onde, quem ou o quê, porém sem valores numéricos representativos de negócios (valores em moeda, quantidades, etc.), apenas descrições. As dimensões permitem filtrar “o como” você irá buscar as informações sobre os fatos, por exemplo: “por lojas”, “por região”, “por mês”, “por produto”, etc.

2.6.2. Esquemas propostos

Sobre o modelo dimensional são conhecidas duas propostas para esquemas de modelagem: o *Star Schema* (Esquema Estrela) e o *Snowflake Schema* (Esquema Floco de Neve). Estes esquemas serão apresentados nas subáreas dessa subseção.

2.6.2.1 Star Schema (Esquema Estrela)

O *star schema* é um modelo sobre uma estrutura física básica em formato estrela, onde componentes periféricos estão todos ligados a um componente central. De acordo com Machado (2007), a composição típica desse esquema possui uma grande entidade central denominada fato (tabela de fatos) e um conjunto de entidades menores denominadas dimensões (tabelas de dimensões), arranjadas ao redor da entidade central.

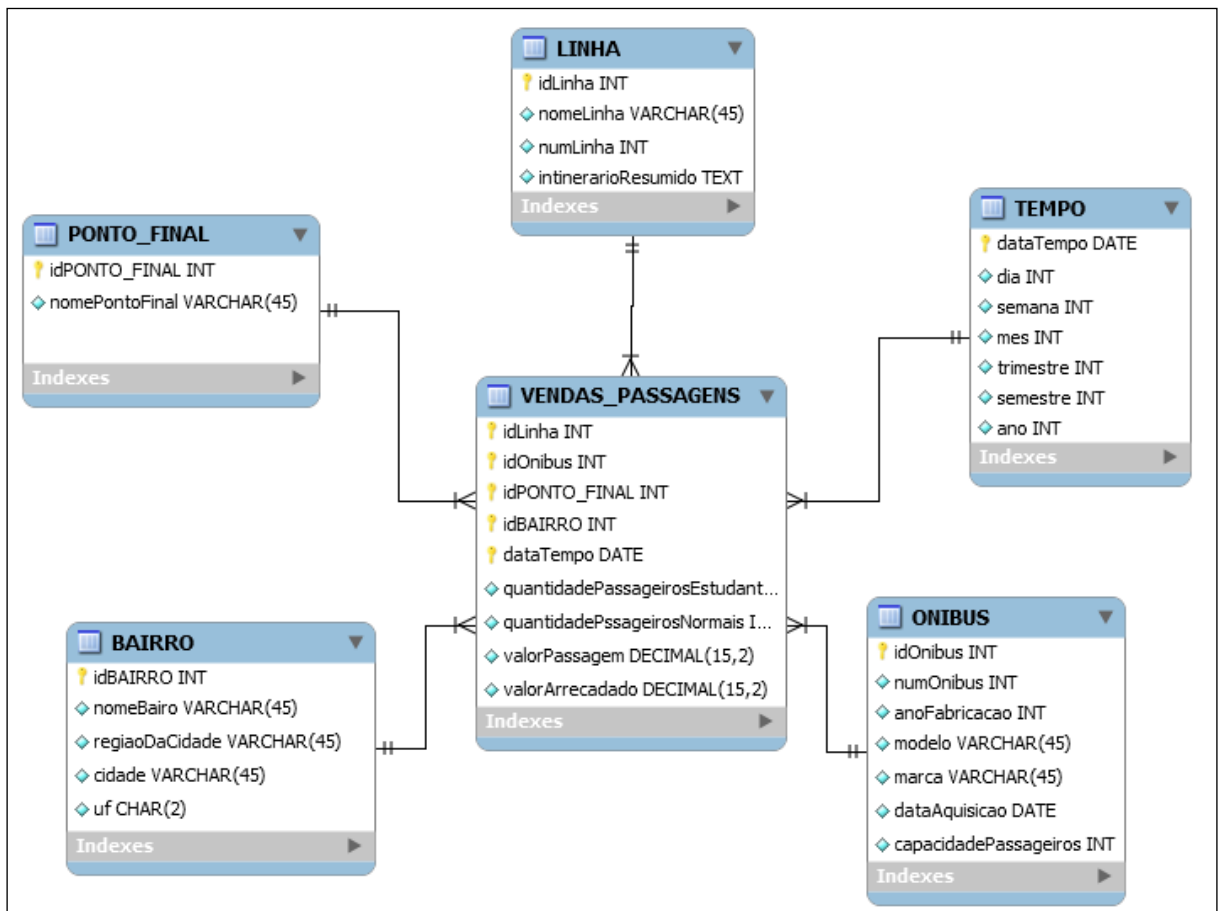


Figura 30 - Modelo dimensional de vendas de passagens de transporte coletivo urbano. Star Schema.

Veja na figura 9, que todas as dimensões estão diretamente relacionadas com a tabela de fatos (VENDAS_PASSAGENS), e o relacionamento é de um para muitos, isto é, uma dimensão participa de muitos fatos. Esse tipo de relacionamento direto permite uma maior agilidade nas consultas, visto que, não é necessário utilizar tantas junções para obter fatos sobre as dimensões.

É notável nesse modelo que não há preocupação com normalização. Deve-se apenas respeitar o preceito de informação rápida, não se preocupando, também, com economia de espaço de armazenamento. Lembrando que em um DW não há inserção de registros via digitação de textos, mas sim por mecanismos programados e de forma única, sem edições de um registro já cadastrado.

2.6.2.2 Snowflake Schema (Esquema Floco de Neve)

O modelo *snowflake* é resultado da decomposição de uma ou mais dimensões que possuem hierarquias entre seus membros [MACHADO, 2007]. Sobre as dimensões diretamente relacionadas à tabela de fatos, foi aplicada a terceira forma normal, o que torna mais fácil o entendimento por desenvolvedores de sistemas OLTP. Observe a figura 10.

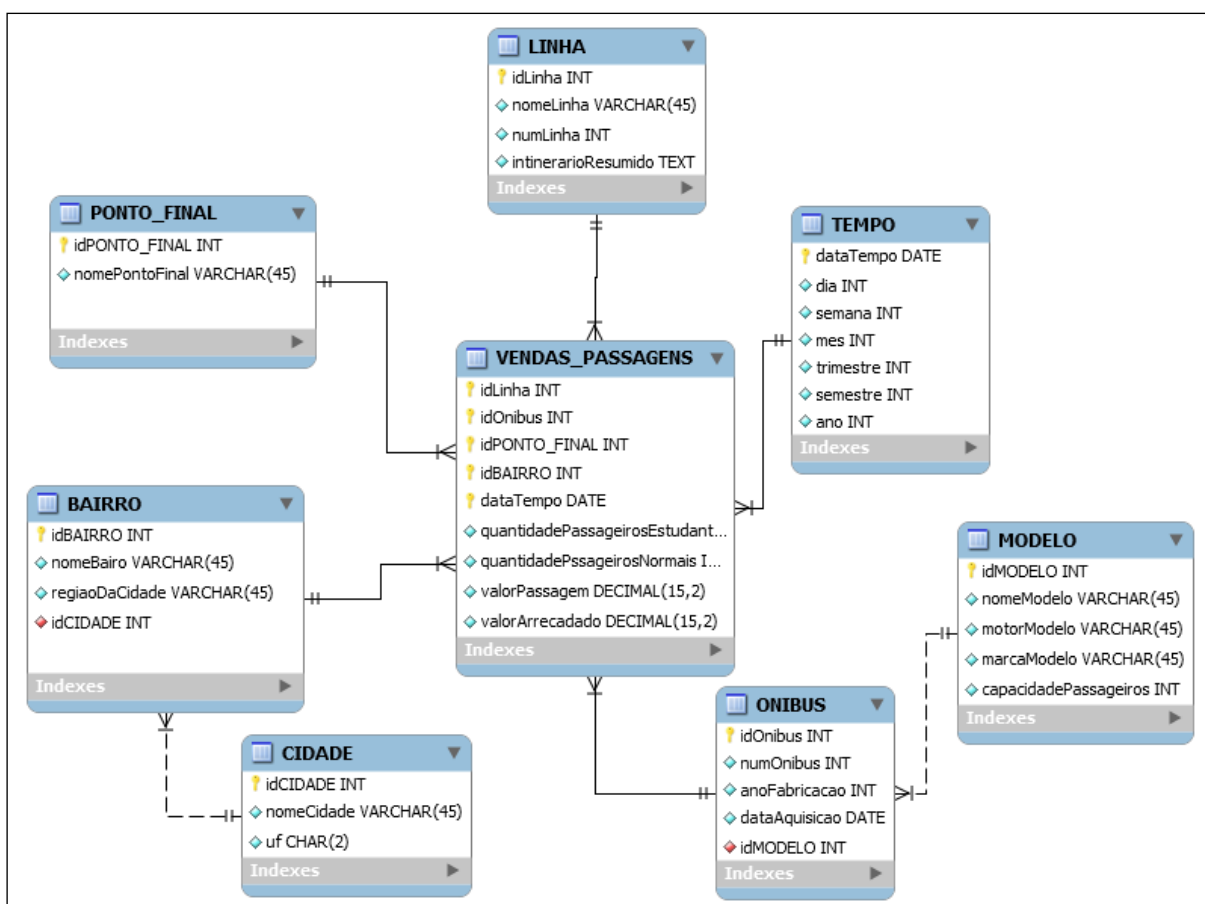


Figura 31 - Modelo dimensional de vendas de passagens de transporte coletivo urbano. Snowflake.

Com a aplicação da terceira forma normal nas entidades dimensões, o modelo evita redundância, porém já foi dito que esse não é o objetivo dos DW, e consultas nesse modelo são mais lentas, visto que, são necessárias mais junções na consulta de fatos de uma dimensão normalizada.

2.6.3. Diferenças entre os modelos ER e Dimensional

Nas subseções anteriores, apresentamos características da modelagem dimensional, que permite uma visão do mundo do negócio e a gestão dos negócios da organização. Já o modelo ER, tradicionalmente conhecido por desenvolvedores de sistemas transacionais, apresenta uma organização com objetivo de modelar os processos operacionais do negócio. O foco primário da modelagem ER é a eliminação de redundâncias e manutenção de consistência sobre diferentes fontes de aplicações. Esse modelo é normalizado, construído respeitando a Terceira Forma Normal e por isso não responde com rapidez questões típicas de consultas de apoio a decisão, já que acabam precisando de 5 ou mais junções de tabelas em uma única consulta.

Segundo Oliveira (2002) “a normalização de dados é uma sequência de etapas sucessivas que, ao final, apresentará um modelo de dados estável com um mínimo de redundância”. A Terceira Forma Normal se dá quando todos os atributos não chave de uma tabela não dependem transitivamente de outro atributo não chave da mesma, se existir uma dependência a tabela deve ser dividida em duas de modo que uma guarde a referência para a outra evitando possíveis redundâncias.

O modelo ER é mais complexo para os usuários que o dimensional, visto que a complexidade se ocorre na necessidade dos usuários realizarem consultas *ad hoc*, e manipularem diretamente os dados sobre esse modelo.

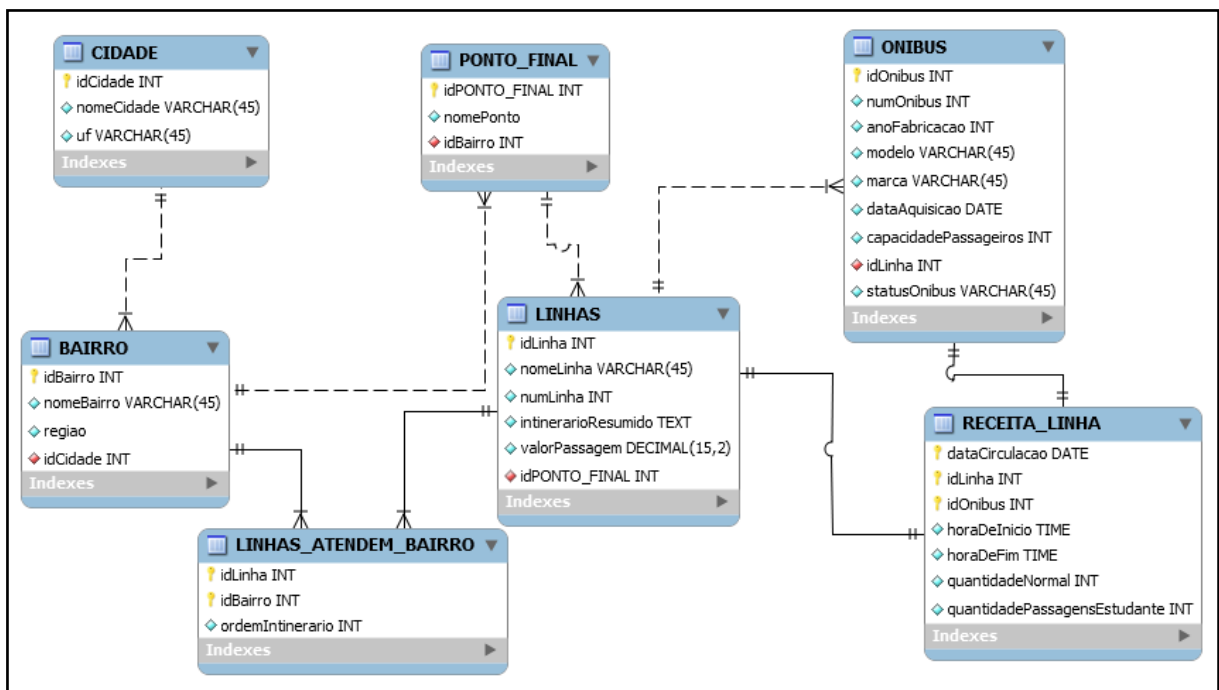


Figura 32 - Modelo ER de gestão operacional de transporte coletivo urbano.

Com a figura 11 acima nós podemos observar de forma exemplificada o funcionamento operacional do sistema de transporte coletivo urbano. E é possível realizar operações como:

- Controlar quais ônibus irão circular em determinadas linhas.
- Controlar o itinerário dessas linhas.
- Controlar a quantidade de passageiros e qual a receita gerada.
- Saber quais ônibus e quais linhas estão relacionados às receitas.

Com a figura 9 é possível obter respostas rápidas para perguntas como:

- Como está a evolução das vendas de passagens ao longo do ano?
- Quais bairros possuem maior participação na receita com vendas de passagens?
- Qual a média de passageiros que circularam por determinadas linhas no último mês? E na última semana?
- Quais linhas apresentarão a maior razão porcentual entre o número de passageiros que circulam e o número de ônibus que rodam na linha no último ano?

Observe que o enfoque da figura 9 é sobre questões que possam envolver tomadas de decisões sobre o negócio, enquanto que a figura 11 apresenta um enfoque operacional.

2.7 On-line Analytical Processing (OLAP)

No dia-a-dia as organizações contam com ferramentas para o controle e operação de seus processos e atividades. Essas ferramentas são, geralmente, sistemas de softwares que suportam transações, que são operações relacionadas às empresas, como pagamentos aos empregados, vendas aos clientes e pagamentos aos fornecedores. Esses sistemas são definidos pela literatura como sistemas OLTP (*On Line Transaction Processing*).

Atualmente a necessidade dos executivos da alta gerência é maior do que controlar a operação do negócio. Existe a necessidade de analisar o andamento, a evolução, as possibilidades de melhoria, obter informações que apoiem a tomada de decisões que objetivem o crescimento do diferencial competitivo da organização. Com base nessas necessidades existe um grupo de ferramentas chamadas OLAP (*On-line Analytical Processing*), Processamento Analítico *On Line*.

2.7.1. Conceitos de OLAP

Segundo Machado (2007) OLAP é um conjunto de ferramentas que possibilita efetuar a exploração dos dados de um DW. A análise realizada através dessas ferramentas permite a descoberta de tendências e cenários, por meio da análise do comportamento determinadas variáveis ao longo do tempo, e obter, a partir dessas tendências e cenários, informações estratégicas sobre o negócio.

Kimball (1998) afirmou que OLAP é um termo inventado com o objetivo de descrever uma abordagem dimensional para o suporte à decisão.

2.7.2. Operações OLAP

O conceito de OLAP envolve, também, algumas operações básicas de conhecimento essencial. Essas operações movimentam a visão dos dados ao longo dos níveis hierárquicos de uma dimensão. Machado (2007) explica as seguintes operações:

Drill down é uma característica que ocorre quando o usuário aumenta o nível de detalhe da informação, diminuindo o nível de granularidade.

Drill up, ou **Roll up**, é o oposto da anterior, ocorre quando o usuário aumenta o nível de granularidade diminuindo o nível de detalhamento da informação.

Drill Across ocorre quando o usuário pula um nível intermediário dentro de uma mesma dimensão. Por exemplo: a dimensão tempo é composta por ano, semestre, trimestre, mês e dia, e o usuário executa um *drill across* quando ele passa de ano direto para semestre ou mês.

Drill Throught ocorre quando o usuário passa de uma informação contida em uma dimensão para outra. Por exemplo: analiso a informação pela dimensão tempo e passo a analisar pela dimensão região.

Slice and Dice são operações para realizar navegação por meio dos dados na visualização de um cubo. Consiste em fatiar os dados permitindo fixar um determinado valor de uma dimensão objetivando diminuir o escopo.

Pivot é a mudança ou troca do ângulo de visão dos dados. Consiste em trocar linhas por colunas em uma tabela ou trocar as dimensões de um gráfico.

2.7.3. Tipos de ferramentas OLAP

As ferramentas OLAP podem ser implementadas de diversas formas. [ARAÚJO *et al.*, 2007 *apud* INMON,1997] classifica essas ferramentas em 5 tipos baseado nas arquiteturas:

- MOLAP (*Multidimensional On Line Analytical Processing*);
- ROLAP (*Relational On Line Processing*);
- HOLAP (*Hybrid On Line Analytical Processing*);
- DOLAP (*Desktop On Line Analytical Processing*);
- WOLAP (*Web On Line Analytical Processing*).

Felber (2005) descreve um pouco cada um desses tipos de ferramentas OLAP.

Uma ferramenta **MOLAP** (*Multidimensional On Line Analytical Processing*) acessa os dados diretamente no banco de dados multidimensional, ou seja, o usuário que faz o trabalho através dessa ferramenta, ele monta e manipula diretamente os dados do cubo⁵ diretamente do servidor.

Em ferramentas **ROLAP** (*Relational On Line Processing*), a consulta é enviada ao servidor de banco de dados relacional e processada no mesmo, mantendo o cubo no servidor, isso permite analisar grandes volumes de dados, entretanto grande quantidade de usuários acessando simultaneamente pode provocar problemas de desempenho no servidor.

As ferramentas sobre uma arquitetura **HOLAP** (*Hybrid On Line Analytical Processing*) possuem uma forma híbrida de acessar os dados, ou seja, uma combinação entre ROLAP e MOLAP. Dessa forma é possível combinar o melhor dos dois mundos, isto é, combinar a alta performance do MOLAP com a alta escalabilidade do ROLAP.

Nas ferramentas sobre a arquitetura **DOLAP** (*Desktop On Line Analytical Processing*) um cliente emite uma consulta para o servidor e obtém como resposta o cubo de informações para ser analisado na estação do cliente. As vantagens dessa arquitetura são: a diminuição do tráfego na rede, a não sobrecarga do servidor de banco de dados e a diminuição de problemas de escalabilidade. Porém as desvantagens podem ser que o cubo de dados seja muito grande a ponto de tornar a análise dos dados muito lenta ou mesmo inviável por conta da configuração da estação cliente.

⁵ Cubo é uma estrutura multidimensional que armazena os dados tornando-os mais fáceis de analisar [FELBER, 2005]. O cubo armazena todas as informações relacionadas a um determinado assunto, de maneira a permitir que sejam montadas várias combinações entre elas, resultando na extração de várias visões sobre o mesmo tema [HOKOMA *et al.*, 2004 *apud* FELBER, 2005].

Por fim, as ferramentas sobre a arquitetura **WOLAP** (*Web On Line Analytical Processing*) são acessadas através de um navegador *web*. Um cliente faz uso do navegador para se comunicar com um servidor HTTP⁶, que por sua vez se utiliza de um *middleware*⁷ responsável pela comunicação com o servidor de banco de dados.

2.8 Data Mining

As técnicas e operações OLAP permitem obter interpretações dos dados existentes, trabalhando os fatos sobre as dimensões e suas hierarquias. O processo de *Data Mining* visa realizar inferências sobre os dados de um DW/DM, tentando “adivinhar” fatos e correlações não explicadas em meio a esses dados.

O conceito de *Data Mining*, garimpagem ou mineração de dados em português, envolve a utilização de algoritmos inteligentes sobre uma amostra dos dados para a detecção de padrões em determinados relacionamentos. Alguns exemplos de padrões são os clássicos vinculados na mídia, como: uma grande porcentagem dos clientes que compram salsichas também compram catchups; uma grande porcentagem dos clientes que compram fraldas também compram cervejas; etc. Esses foram exemplos de padrões em comportamento dos clientes, mas é possível identificar outros padrões como, por exemplo, estilos de ações fraudulentas em cartão de crédito ou seguradoras [BARBIERE, 2001].

Segundo Moss e Atre (2003), a aplicação de mineração de dados pode então usar uma sofisticada mistura de componentes clássicos e avançados, como a inteligência artificial, reconhecimento de padrões, bases de dados, estatísticas tradicionais e gráficas para apresentar relações ocultas e padrões encontrados no repositório de dados da organização.

Ainda de acordo com Moss e Atre (2003), a mineração de dados é a análise dos dados com a intenção de descobrir jóias de informações escondidas na vasta quantidade de dados que foi capturado no curso normal de funcionamento do negócio. A mineração de dados é diferente da análise estatística convencional, como indicado na tabela 1.

Tabela 1- Comparativo entre Análise Estatística e Data Mining [MOSS e ATRE, 2003].

Análise Estatística	Data Mining
Estatísticas normalmente começam com uma hipótese.	<i>Data mining</i> não necessita de uma hipótese.
Os estatísticos têm de desenvolver suas próprias equações para coincidir com a sua hipótese.	Algoritmos de <i>Data mining</i> podem desenvolver automaticamente suas equações.
Análise estatística utiliza apenas dados numéricos.	<i>Data mining</i> pode utilizar diferentes tipos de dados (texto, voz), não apenas dados numéricos.
Estatísticos podem encontrar e filtrar dados sujos durante suas análises.	<i>Data mining</i> dependerá da limpeza, dos dados bem documentados.
Estatísticos interpretam seus próprios resultados e transmitem esses resultados para os gerentes de negócios e executivos.	Resultados de mineração de dados não são fáceis de interpretar. A estatística ainda deve estar envolvida na análise dos

⁶ HTTP (acrônimo do inglês, *Hypertext Transfer Protocol*, Protocolo de Transferência de Hipertexto) é um protocolo de comunicação (na camada de aplicação segundo o Modelo OSI) utilizado para sistemas de informação de hipermídia distribuídos e colaborativos [BERNERS-LEE *et al.*, 1996].

⁷ Middleware é um programa de computador que faz a mediação entre outros softwares.

	resultados da mineração dos dados e transmitir os resultados aos gestores de empresas e executivos de negócios.
--	---

2.9 Projeto de um Data Warehouse ou Data Mart

Barbieri (2001) apresenta em sua obra o que para ele são as principais fases para projetos de DW/DM:

- 1) Planejamento
- 2) Levantamento das necessidades
- 3) Modelagem dimensional
- 4) Projeto físico de banco de dados
- 5) Projeto de transformação
- 6) Desenvolvimento de aplicações
- 7) Validação e testes
- 8) Treinamento
- 9) Implantação

Barbieri (2001) considera as cinco primeiras fases as mais críticas para um projeto de DW, e as quatro últimas passíveis de muitos problemas caso não sejam bem planejadas e realizadas. Seguindo a ideia de Barbieri (2001), será apresentado nas próximas subseções os principais objetivos e preocupações de cada uma das fases citadas.

2.9.1. Planejamento

Na fase de planejamento, os principais objetivos são: definir o escopo do projeto, com um olhar mais atento para as áreas críticas da empresa e para as necessidades mais urgentes e importantes; definição da abordagem para implementação do DW/DM; planejar as integrações; e definir uma arquitetura tecnológica.

Para a definição do escopo a principal preocupação é manter o foco no negócio, por esse motivo é interessante definir uma metodologia comunicativa e formal, como JAD⁸ (*Joint Application Design*), sugerido por Barbieri (2001).

Realizar um planejamento de integração é de extrema importância e consiste em analisar e identificar elos entre as necessidades atuais e as futuras, permitindo pensar em outras áreas do negócio, a fim de diminuir o retrabalho e minimizar os traumas com integrações futuras entre *Data Marts*.

É fundamental que os componentes da arquitetura sejam definidos nessa fase de planejamento antes do início do projeto, pois fatores como performance e disponibilidade estão diretamente ligados a características do DW, como granularidade, por exemplo, além disso determinados níveis de serviços e compromissos podem ser definidos baseados nesses fatores.

⁸ *Joint Application Development*, ou *Joint Application Design* (JAD) é uma metodologia que acelera o projeto de sistemas. Guiados por um líder de reunião, usuários e analistas projetam o sistema juntos, em sessões de grupo estruturadas. JAD utiliza a criatividade e o trabalho em equipe de dinâmica de grupo para definir o ponto de vista dos usuários sobre o sistema, desde os objetivos e aplicações do sistema até a geração de telas e projetos de relatórios [WIKIPEDIA, 2009].

2.9.2. Levantamento de necessidades

Para essa fase de levantamento de necessidades o objetivo é basicamente identificar dois modelos: Dimensional e o modelo Fonte de Dados. O modelo Dimensional é obtido através da garimpagem dos dados a serem modelados em seus vários níveis de sumarização e detalhes, para alcançar os objetivos de suporte à decisão. Já o modelo Fonte de Dados, como o próprio nome sugere, deve registrar os blocos conceituais, descrições e formas atuais de armazenamento dos dados existentes nas fontes de informações. É importante observar nessa fase a qualidade, a integridade e a duração histórica dos dados fonte.

2.9.3. Projeto lógico

Baseado na ideia de Barbieri (2001), na fase do Projeto Lógico deve-se analisar e formalizar a definição das fontes de dados levantadas na fase anterior, além de definir a granularidade, as dimensões, as medidas e a construção do modelo dimensional, com a modelagem da entidade fato e as entidades dimensões, relacionamentos, indexação, atributos de entidades e implantação de regras.

2.9.4. Projeto físico

Na fase de projeto físico a equipe projetista deverá considerar o uso de um SGBD⁹ Relacional ou de um SGBD Dimensional. Além do tipo, também é importante definir qual SGBD adquirir, se adquirir uma solução livre e gratuita ou uma solução proprietária, conhecida no mercado. A sugestão é que a escolha seja definida pelas características de performance e disponibilidade desejadas, além do fator custo benefício.

Após a escolha do tipo e do SGBD, é hora de escrever ou gerar os scripts de criação compatíveis com o SGBD escolhido.

Ainda nessa fase envolvemos o projeto de *Extract Transform Load (ETL)*, ou em português, *Extração Transformação e Carga*. Nesse projeto serão definidos os processos de transformação do modelo Fonte para o modelo Dimensional. No conceito de extração e tratamento, os dados poderão ser filtrados, integrados, condensados, convertidos e/ou derivados.

2.9.5. Desenvolvimento de aplicações

O desenvolvimento de aplicações para fazer uso dos dados armazenados no DW/DM é, praticamente, um novo projeto dentro do projeto de DW/DM, porém existe no mercado um conjunto de ferramentas dedicadas ao desenvolvimento de aplicações OLAP, bem como, existem ferramentas específicas para aplicações de *Data Mining*. Sobre essas ferramentas são construídas aplicações de acordo com as necessidades do cliente. Citarei algumas ferramentas ou suítes de ferramentas na próxima seção.

2.9.6. Validação e testes

Nessa fase o sistema deve ser testado, sendo simulados testes com o máximo possível de volumes de dados e de processamento. Barbieri (2001) sugere que o sistema

⁹ SGBD é o acrônimo para Sistema de Gerenciamento de Banco de Dados. O principal objetivo é retirar da aplicação cliente a responsabilidade de gerenciar o acesso, manipulação e organização dos dados. O SGBD disponibiliza uma interface para que os seus clientes possam incluir, alterar ou consultar dados [QUEIROZ, 2009].

deve ser liberado inicialmente a um grupo restrito de usuários e após a análise de feedbacks, entregue ao ambiente produtivo.

2.9.7. Implantação

Na implantação estão contidas as atividades de treinamento aos devidos usuários do produto, usuários voltados às atividades do negócio, gerentes das áreas envolvidas, bem como as atividades de acompanhamento do uso das aplicações disponibilizadas.

Ao longo de todo o projeto deve ser construído um diretório de metadados do projeto, nada mais do que um grande dicionário de dados amistoso, que descreve os dados do modelo Fonte, das suas transformações, do modelo Dimensional, de suas formas, de seus acessos e de sua disponibilização.

2.10 Ferramentas do mercado

Essa seção apresenta algumas empresas e suas ferramentas de BI conhecidas no mercado, a começar pela *MicroStrategy* [MICROSTRATEGY, 2009] com sua plataforma de software para BI, o *MicroStrategy 9*. Essa empresa detém uma grande parte do mercado no que diz respeito a empresas mundialmente conhecidas. O objetivo do *MicroStrategy 9* é atingir desde aplicações pequenas até grandes operações em âmbito corporativo.

A SAP [SAP, 2009] é outra fornecedora de soluções para BI, essa empresa apresenta o *SAP BusinessObjects Intelligence Platform*, é uma plataforma BI que envolve softwares para a detecção e distribuição de informações, gerenciamento de informações; além de consultas, elaboração de relatórios e análises.

A ORACLE e a Microsoft também não estão de fora desse mercado. A ORACLE [ORACLE, 2009] possui a *ORACLE Business Intelligence Suite Enterprise Edition* com ferramentas de administração de DW, aplicações de suporte à decisão em tempo real, entre outras coisas. A Microsoft [MICROSOFT, 2009] integrou ao SQL Server 2008 alguns módulos como o *Analysis Services*, *Integration Services*, *Reporting Services*, entre outros. Além disso, a Microsoft apresenta uma integração dessas ferramentas com o Microsoft Office 2007, dando aos usuários uma impressão de manipular informações em ferramentas comuns do seu dia-a-dia.

Uma ferramenta interessante é a *QlikView*, atualmente na versão 8.5 é uma pequena plataforma de desenvolvimento para visualização e análise dados de BI. É desenvolvida pela *QlikTech* [QLIKVIEW, 2009], e a missão dessa empresa é simplificar a análise para todos, permitindo o desenvolvimento rápido, claro, divertido e inteligente, ainda com ganhos de performance. A maior propaganda do *QlikView* é desenvolvimento rápido em torno de dias, não meses, o que agrada a clientes pela idéia de obter um retorno mais rápido.

Uma plataforma *Open Source*¹⁰ é o *Pentaho BI* [PENTAHO, 2009] que provê a arquitetura e infra-estrutura necessária para a construção de soluções para problemas de BI. Esse projeto envolve outros projetos *open sources* como o *Kettle*, que é uma ferramenta de integração de dados (ETL) orientada a *metadados*, o *Mondrian*, que é um servidor OLAP escrito em Java, que permite a análise de grande escala de dados armazenados, sem a necessidade de escrever comandos SQL, o *Pentaho Reporting*, que é uma coleção de outros projetos com foco na criação, geração e distribuição de relatórios ricos e sofisticados, e o *Pentaho Data Mining*, que é baseado no projeto

¹⁰ Open Source refere-se a software de código fonte aberto à comunidade para modificação e/ou melhorias colaborativas.

Weka, uma ferramenta de aprendizado de máquinas e de mineração de dados. Esse projeto pode ser encontrado gratuito na comunidade *Pentaho*, mas também possui uma versão comercializada a versão *Enterprise Edition*, que envolve treinamentos, suporte entre outras coisas.

3 ESTUDO DE CASO

Nesse capítulo estarei apresentando a construção de estudo de caso que tenho como proposta para a realização de um projeto futuro de um DW sobre sistemas de transportes coletivos urbanos. A idéia é aprender e mostrar como se inicia um projeto de DW, apresentando algumas técnicas e alguns mitos sobre o início do processo. Não é o objetivo apresentar um projeto de DW/DM completo ou avançado, passando por todas as fases citadas no capítulo anterior, mas apenas um estudo de caso que amplifique o conhecimento introdutório sobre projetos de DW/DM apresentando alguns conceitos e noções sobre como definir e identificar os fatos e as dimensões do negócio.

3.1 Entendendo o problema

Em um projeto de DW/DM, a fase de planejamento tem seu início com a análise do cliente. É importante traçar o perfil do cliente e do negócio, o objetivo é visualizar possíveis temas que possam compor o projeto e compor futuras necessidades do cliente. Para identificar o perfil do cliente e do negócio é importante obter respostas para algumas das perguntas abaixo:

- Empresa pública ou privada?
- Qual a área de negócio?
- Qual a área geográfica de atuação?
- Existe concorrência?
- Qual a posição diante da concorrência?
- Qual o perfil dos executivos da empresa (são conservadores, inovadores ou intermediários)?
- Tem algum tipo de associação com empresas estrangeiras?
- É uma empresa familiar?
- Há quantos anos está no mercado?
- Possui sistemas de gestão?
- Quais tecnologias (hardware, software, capacitação do pessoal, etc.) a empresa tem acesso?
- Possui um departamento de TI?
- Se possuir, como esse departamento interfere nos negócios e nas decisões?
- Etc.

Para exemplificar o nosso estudo de caso nosso cliente será um cliente fictício, uma secretaria de transportes urbanos de uma prefeitura. É um órgão público do governo municipal, um órgão pouco informatizado, que possui um sistema de gestão de controle de passagens sobre as linhas de ônibus que circulam no município. O órgão não possui um departamento de TI interno próprio, mas a prefeitura do município possui um departamento TI que em nada interfere nos negócios, apenas funciona como controle e manutenção das redes internas e com o bom funcionamento dos softwares da prefeitura.

Com o perfil traçado, o próximo passo é identificar quais são as necessidades executivas desta empresa para o projeto, essas necessidades irão permitir analisar a viabilidade do projeto e colaborar para a definição do projeto lógico.

As necessidades identificadas com o cliente foram:

- 1) O secretário de transportes do município precisa acompanhar a evolução do volume de passageiros que viaja pelas linhas de transportes coletivos das empresas, nos bairros por onde circulam os ônibus dessas linhas.
- 2) Esse acompanhamento precisa ser feito ao longo dos meses, trimestres e anos.
- 3) Também é necessário avaliar se os passageiros são estudantes ou não, e a ocorrência de passagens gratuitas devido a um sistema de integração temporal, onde os passageiros circulam por mais de um ônibus e linha, em um intervalo de tempo, pagando apenas a primeira passagem, ou seja, a segunda é gratuita.
- 4) O volume de estudantes ou pessoas comuns que circularam nos últimos meses por linhas que passam em determinados bairros.
- 5) O avanço no volume de passagens gratuitas para determinadas linhas ou determinadas empresas nos últimos meses.
- 6) É importante, também, identificar quais ônibus que mais circularam em um determinado período, por quais linhas e a quantidade de passageiros carregados por ele nesse período, podendo filtrar essas informações pela idade do ônibus.

Essas são algumas das necessidades apresentadas para esse problema. Na próxima seção vamos fazer uma análise das necessidades apresentadas e iniciar o nosso modelo lógico dimensional.

3.2 Analisando as necessidades e modelando o DW/DM

Uma pergunta comum para se fazer nesse momento é: “O que faço agora? Começo analisando o modelo atual, ou modelo o DM pelas necessidades apresentadas?”. Digamos que o cliente forneceu o modelo ER do banco de dados do seu sistema. Isso nos ajudará a entender se estamos trabalhando com os fatos corretos e se existem mais fatos não atingidos pelas necessidades apresentadas pelo cliente, e também se existem dimensões não percebidas. Porém, segundo Machado (2007) é uma inverdade afirmar que se deve transformar o modelo ER do sistema transacional em um modelo de dados estrela utilizando uma ferramenta CASE¹¹. Então a estratégia será criar o modelo dimensional estrela através da análise das necessidades, posteriormente acessar o modelo ER do banco de dados do sistema transacional de controle do transporte coletivo urbano.

3.2.1. Identificando os fatos e as dimensões

A primeira etapa desse processo de análise das necessidades e construção da modelagem é iniciada pela identificação dos fatos, ou inicialmente, pelo menos um fato. A partir desse primeiro fato identificado, podemos identificar também as quatro dimensões básicas, as que respondem a perguntas como [MACHADO, 2007]:

- **Onde** acontece o fato?
- **Quando** acontece o fato?

¹¹ Ferramenta CASE (do inglês *Computer-Aided Software Engineering*), que significa Engenharia de Software Auxiliada por Computador é uma classificação que abrange todas ferramentas baseada em computadores que auxiliam atividades de engenharia de software, desde análise de requisitos e modelagem até programação e testes [SOMMERVILLE, 2009].

- **Quem** realiza o fato?
- **O que** acontece no fato?

A primeira solicitação do cliente foi: “*O secretário de transportes do município precisa acompanhar a evolução do volume de passageiros que viajam pelas linhas de transportes coletivos das empresas, nos bairros por onde circulam os ônibus dessas linhas*”. Para encontrar o fato nessa solicitação deve-se procurar responder perguntas como:

- Qual o fato desta necessidade?
- O que dá a ideia de ação?
- O que se caracteriza por poder ser medido ou possuir medidas?
- A evolução do volume de passageiros é um indicador de negócio?

Nessa primeira solicitação palavras que passam a ideia de ação são “passageiros que viajam”. Essas viagens podem ser medidas em quantidades, em valores numéricos. E a evolução do volume de passageiros pode sim se caracterizar como indicadores de negócios. Por tanto podemos definir como um fato as VIAGENS DE PASSAGEIROS.

Com um primeiro fato identificado é hora de identificar inicialmente as quatro dimensões básicas, as dimensões onde, quando, quem e o quê.

3.2.1.1 Dimensão Onde

Já nessa primeira necessidade apontada nós temos a resposta para a pergunta “Onde?”. A resposta se refere aos locais por onde as linhas circulam, nos bairros do município. Podem existir outras dimensões onde, por exemplo, ônibus pode ser um tipo de onde. Podem existir ilimitadas possibilidades de onde, quando, quem e o quê, relativos a um mesmo fato.

3.2.1.2 Dimensão O quê

As linhas de transporte são o quê do problema, elas são o que deve ser acompanhado nos fatos. É importante acompanhar se cresce ou não o número de passageiros circulando pelas linhas que passam por determinados bairros. As linhas podem ter diferentes evoluções. Por exemplo, as linhas que circulam pelos bairros Mangabeira e Bancários possuem números de passageiros próximos dos números das linhas que circulam pelos bairros de Mandacaru e 13 de Maio?

3.2.1.3 Dimensão Quem

Responder quem viaja nesse caso não é a resposta ideal, pois veremos que passageiros não é reportado nesse problema como uma entidade, mas de acordo com a necessidade 4, podemos considerar o tipo de passageiro como uma entidade. Porém, essa entidade será considerada na subseção 3.2.3. Então a pergunta pode ser “quem transporta os passageiros?”, alguns poderiam dizer ônibus, essa pode ser uma resposta, mas outra resposta pode ser empresa, que é dona dos ônibus e detentora de linhas de transporte. Por tanto, temos duas dimensões quem, ônibus e empresa.

3.2.1.4 Dimensão Quando

A necessidade 2 (dois) ajuda a identificar a dimensão quando ou dimensão tempo. Essa necessidade apresenta os atributos mês, trimestre e ano. Porém como é solicitado fazer o acompanhamento ao longo dos meses é necessário identificar o dia em

que o fato ocorreu para ser delimitado um marco no mês, por exemplo, para que todas as comparações sejam feitas com referência aos dias 15 (quinze) de cada mês.

Foram identificadas nessa análise baseada nos dois primeiros requisitos 1 (um) fato e as dimensões básicas. Veja na figura 12 abaixo.

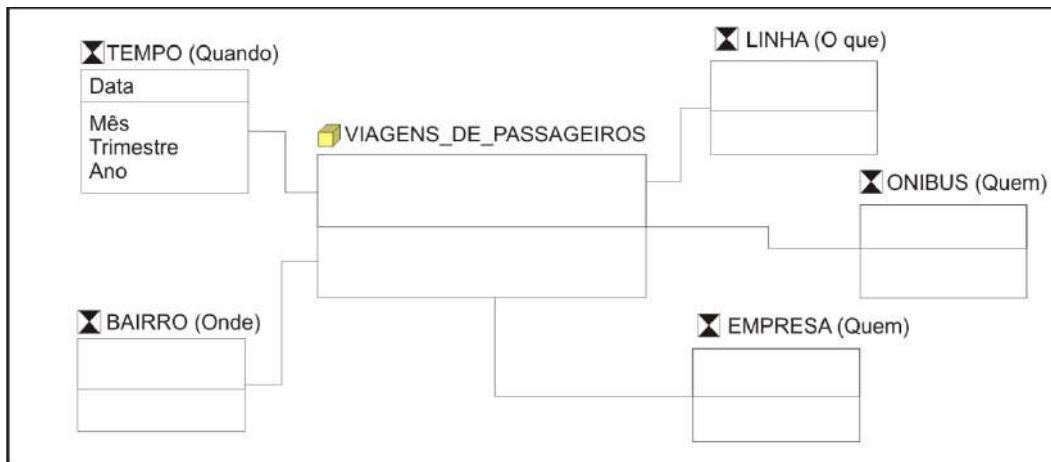


Figura 33 - Fato e dimensões básicas do problema.

3.2.2. Resolvendo as chaves das entidades dimensões e da entidade fato

Observe na figura 12 que tanto as dimensões quanto os fatos estão sem as chaves que determinam a integridade dos dados. Propositamente não foram colocadas as chaves por existirem de dúvidas comuns, de iniciantes, sobre elas. Levando em consideração que os dados serão extraídos da base de dados do sistema transacional, as chaves das entidades lá encontradas serão as mesmas chaves das dimensões nesse novo modelo? A resposta é não. O motivo é simples, o DW/DM não é transacional, isto é, ele não sofre alterações nos dados cadastrados ao longo do tempo, os dados no DW/DM são apenas incrementados. Exemplificando melhor, em um sistema transacional um bairro pode ter o código 120 hoje e amanhã esse código pode ser alterado, ou mesmo o bairro pode nem mais existir. Já no DW/DM não pode ocorrer tal alteração e o bairro continuará existindo, mesmo não mais existindo no sistema transacional. Por isso as chaves das entidades dimensões devem ser seqüenciais e geradas automaticamente a partir do processo ETL. A chave da entidade fato é uma chave composta pelas chaves das dimensões participantes do fato. Observe agora a figura 13.

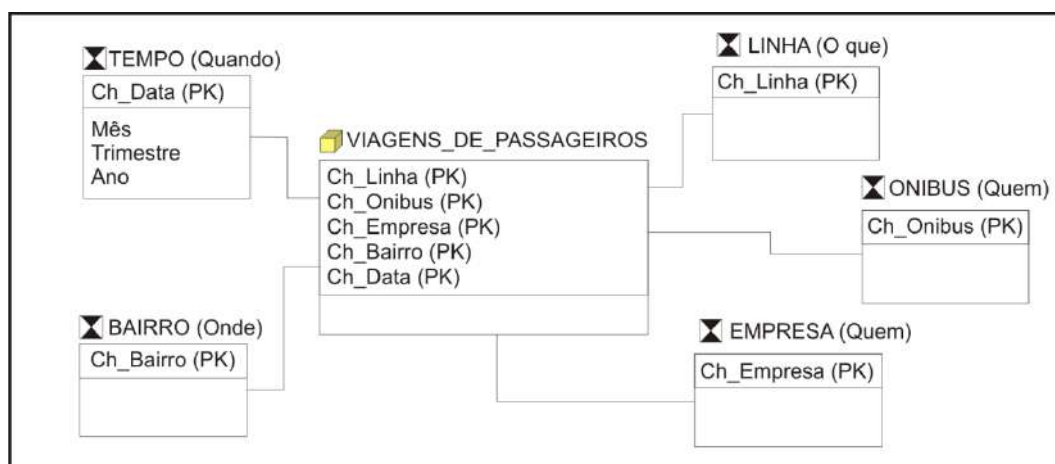


Figura 34 - Fato e dimensões básicas com chaves

Os atributos das dimensões serão identificados mais adiante com a análise do modelo ER do sistema transacional.

3.2.3. Identificando dimensões mascaradas

Segundo Machado (2007) “dimensões mascaradas são aquelas que não são implementadas como entidades, e sim com atributos de um fato”. As dimensões mascaradas são utilizadas quando uma entidade dimensão não possui número significativo de ocorrências.

Nas necessidades 3 (três), 4 (quatro) e 5 (cinco) apresentam duas novas entidades dimensões, a dimensão tipo passageiro e a dimensão tipo passagem. A dimensão tipo passageiro, de acordo com a necessidade passada, apresenta apenas o tipo estudante e o não estudante, apesar de que na realidade do cliente do problema existem ainda o tipo idoso e o tipo deficiente, porém esses dois últimos tipos não são registrados no sistema transacional, por tanto, são apenas duas ocorrências da entidade e podemos colocá-la como dimensão mascarada, isto é, um atributo “tipo passageiro” na entidade fato. O mesmo ocorre para a dimensão tipo passagem, as passagens só podem ser de estudante, inteira e gratuita, e pela pouca ocorrência também vira uma entidade mascarada como atributo “tipo passagem” do fato.

Observe que, como no DW/DM os usuários não inserem os dados diretamente na base, mas os dados são inseridos de forma programada através de um processo ETL, não teremos problemas de inconsistências por escrita errada dos dados, por tanto não há problemas em utilizar as dimensões mascaradas, elas melhoram o desempenho do DW/DM.

Veja na figura 14 as dimensões mascaradas e observe que elas também formam a chave composta da entidade fato.

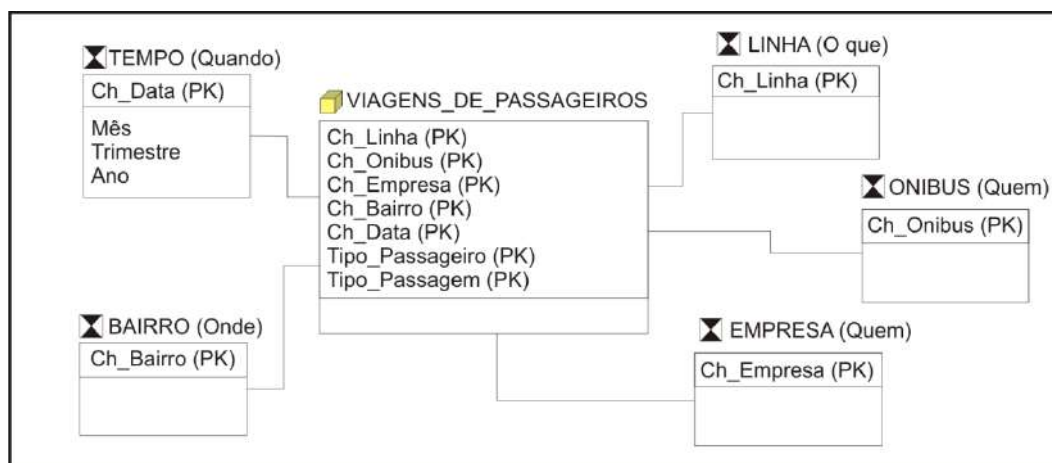


Figura 35 - Fatos e dimensões básicas e mascaradas

Basicamente atingimos todas as necessidades citadas na seção 3.1 deste capítulo, então entraremos em uma segunda fase da análise e modelagem do DW/DM. É hora de fazer a análise baseada no modelo ER do sistema transacional.

3.2.4. Analisando o modelo ER do sistema transacional

Segundo Machado (2007, pág. 179) “todo relacionamento muitos para muitos em um modelo transacional está na realidade representando um fato na vida real”. Ele

quis dizes que as tabelas associativas e as agregações representam fatos do negócio. Observe a figura 15 abaixo.

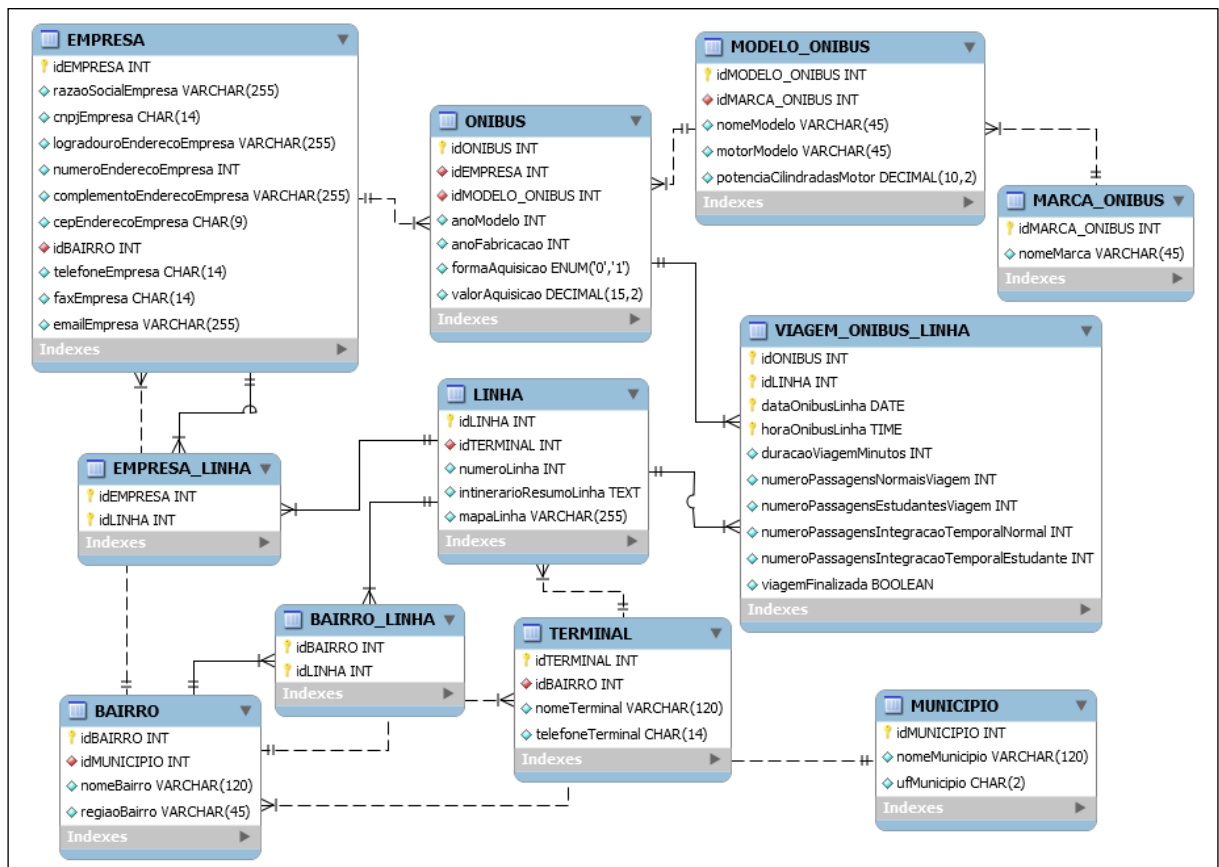


Figura 36 - Diagrama ER do software de acompanhamento de passagens de um sistema de transporte coletivo urbano.

Veja que na figura 15 é possível identificar três tabelas associativas, tabelas de relacionamentos muitos para muitos. As tabelas identificadas são: EMPRESA_LINHA, BAIRRO_LINHA e VIAGEM_ONIBUS_LINHA. Observe, também, como essas tabelas representam fatos do negócio. Uma empresa pode possuir várias linhas de ônibus e uma linha pode pertencer a mais de uma empresa. Isto é um fato. Uma linha circula por vários bairros e por um mesmo bairro circulam várias linhas. Isto também é um fato. Porém o mais representativo é o fato de que os ônibus podem realizar várias viagens por várias linhas diferentes e por uma mesma linha pode viajar vários ônibus. O que torna esse fato mais representativo é a ocorrência de medidas sobre ele, por exemplo, a duração das viagens realizadas por todos os ônibus em determinada linha e a quantidade de passageiros, estudantes ou não, que participou dessas viagens. Essa associação entre ônibus e linha é basicamente o fato trabalhado no nosso modelo estrela até o momento. É importante analisar e comparar esse fato no modelo ER e no modelo estrela.

3.2.4.1 Revendo a granularidade

Observando a tabela VIAGEM_ONIBUS_LINHA no modelo ER e a entidade fato “Viagens de Passageiros” no modelo dimensional, nota-se uma diferença de granularidade, na tabela VIAGEM_ONIBUS_LINHA os registros são realizados a cada viagem, isto é, ao longo de um dia várias viagens ocorrem e é adicionada uma linha na

tabela para cada nova viagem. Já no modelo dimensional são sumarizados os fatos ocorridos ao longo de um dia (uma data), isto é, as ocorrências de um dia são sumarizadas em uma “única” linha no banco. Observe as tabelas 2 e 3 que apresentam diferentes níveis de granularidade.

Tabela 2 - Exemplo de ocorrências em menor granularidade.

EMPRESA	ÔNIBUS	LINHA	DATA	HORA	DUR	P	TP	Q
Mandacaruen se	0446	504	25/11/2009	07:00	1,5h	NE	NR	73
Mandacaruen se	0446	504	25/11/2009	07:00	1,5h	ES	NR	32
Mandacaruen se	0446	504	25/11/2009	07:00	1,5h	NE	GR	3
Mandacaruen se	0446	504	25/11/2009	07:00	1,5h	ES	GR	4
Mandacaruen se	0446	504	25/11/2009	08:45	1,3h	NE	NR	57
Mandacaruen se	0446	504	25/11/2009	08:45	1,3h	ES	NR	28
Mandacaruen se	0446	504	25/11/2009	08:45	1,3h	NE	GR	6
Mandacaruen se	0446	504	25/11/2009	08:45	1,3h	ES	GR	5

Tabela 3 - Exemplo de ocorrências em maior granularidade.

EMPRESA	ÔNIBUS	LINHA	DATA	P	TP	Q
Mandacaruense	0446	504	25/11/2009	NE	NR	586
Mandacaruense	0446	504	25/11/2009	ES	NR	417
Mandacaruense	0446	504	25/11/2009	NE	GR	40
Mandacaruense	0446	504	25/11/2009	ES	GR	33
Mandacaruense	0446	504	26/11/2009	NE	NR	572
Mandacaruense	0446	504	26/11/2009	ES	NR	397
Mandacaruense	0446	504	26/11/2009	NE	GR	31
Mandacaruense	0446	504	26/11/2009	ES	GR	36

Nas tabelas 2 e 3 existem as colunas **P**, **TP** e **Q**. A coluna **P** significa Passageiros e como registros essas colunas possuem NE e ES, NE significa Não-Estudante e ES Estudante. A coluna **TP** significa Tipo de Passagem, que pode ser NR (Normal) e GR (Gratuita). A coluna **Q** representa a Quantidade e passageiros. Na tabela 2 existe ainda a coluna **DUR** que representa a Duração da viagem, que na tabela 4 está representada como **D. MÉDIA** (Duração Média das viagens).

Observe que na tabela 3 o número de ocorrências de viagens de uma linha de uma mesma empresa em um mesmo dia é menor do que na tabela 2, isso por que na tabela 2 a granularidade é menor e o nível de detalhe é maior, podendo obter dados

sobre cada viagem realizada por um ônibus de uma determinada empresa por uma determinada linha em um mesmo dia.

Para esse cliente, não é o objetivo saber tão detalhadamente como se comportaram as viagens, porém se a preocupação dele fosse analisar as viagens sobre o ponto de vista da duração, teríamos que diminuir a granularidade? A resposta é não. Nós poderíamos armazenar para cada dia a duração média das viagens (veja tabela 4). Porém, se a preocupação existisse em analisar a variação dessa duração em relação ao número de passageiros ao longo dos dias, provavelmente teríamos a granularidade apresentada na tabela 2.

Tabela 4 - Exemplo de ocorrências em maior granularidade com média de variação na duração das viagens.

EMPRESA	ÔNIBUS	LINHA	DATA	D. MÉDIA	P	TP	Q
Mandacaruese	0446	504	25/11/2009	1,4h	NE	NR	586
Mandacaruese	0446	504	25/11/2009	1,4h	ES	NR	417
Mandacaruese	0446	504	25/11/2009	1,4h	NE	GR	40
Mandacaruese	0446	504	25/11/2009	1,4h	ES	GR	33
Mandacaruese	0446	504	26/11/2009	1,3h	NE	NR	572
Mandacaruese	0446	504	26/11/2009	1,3h	ES	NR	397
Mandacaruese	0446	504	26/11/2009	1,3h	NE	GR	31
Mandacaruese	0446	504	26/11/2009	1,3h	ES	GR	36

3.2.4.2 Analisando atributos para dimensões e medidas dos fatos

Baseado nos atributos das tabelas do modelo ER do sistema transacional, vamos observar os atributos que irão compor as entidades dimensões. Para a entidade dimensão Tempo, nós já possuímos os atributos necessários, visto que a maioria deles não é extraída no modelo ER, mas sim gerada no processo ETL. Porém, para as demais dimensões os atributos são diretamente compatíveis, o que não significa que deva apenas copiar o mesmo atributo do modelo ER.

Para a entidade dimensão Empresa podemos definir os atributos razão social e CNPJ, esses atributos bastam para descrever essa entidade. Para a entidade dimensão Linha são definidos os atributos número da linha, título da linha, terminal da linha e o resumo do itinerário. Para as entidades dimensões Bairro e Ônibus, vamos fazer uma análise mais aprofundada, visto que no modelo ER essas entidades possuem uma hierarquia. Veja essas hierarquias na figura 16.

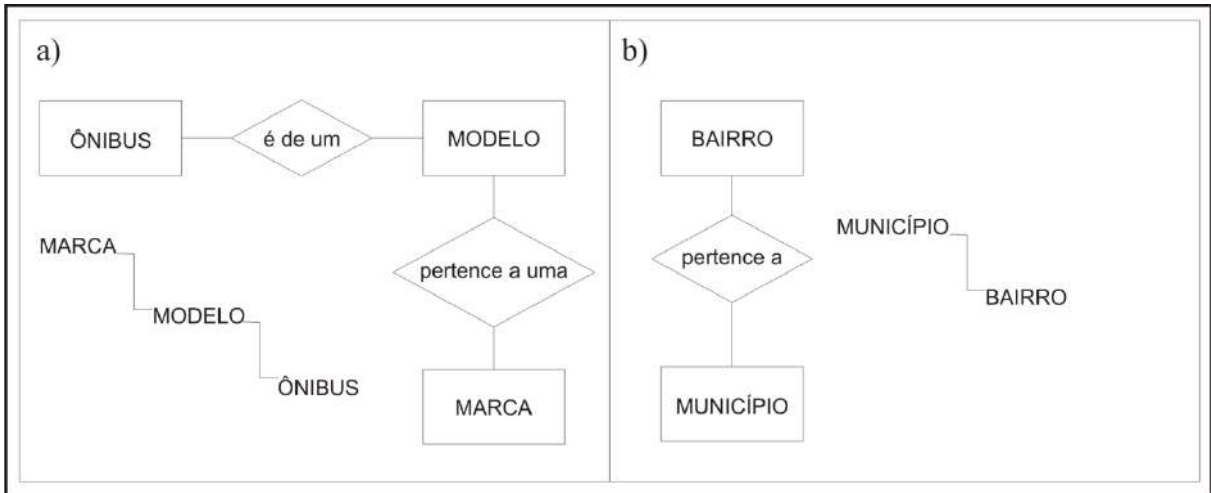


Figura 37 - Hierarquias referentes às entidades dimensões Ônibus (a) e Bairro (b).

Essas hierarquias sugerem a utilização de normalização no modelo dimensional, de modo a atingir o modelo *snowflake*, visto que esse modelo minimiza a redundância dos dados. Mas redundância não é a nossa preocupação. A preocupação é atingir as necessidades e garantir o desempenho esperado para o DW/DM. Por isso esses níveis hierárquicos podem se transformar em atributos na entidade dimensão. Por tanto, para a entidade dimensão Ônibus, os atributos são número do ônibus, modelo, marca, ano de fabricação (a idade do ônibus participa das necessidades) e motor. Para a entidade bairro os atributos definidos são o nome do bairro, a região da cidade onde se localiza o bairro, o município e a unidade federativa do município.

Como medida do fato foi identificada a quantidade de passageiros nas viagens. Essa quantidade ficará detalhada por tipos de passageiros (coluna P na tabela 3), por tipos de passagens (coluna TP na tabela 3). Veja na figura 17 como ficou o nosso modelo dimensional final.

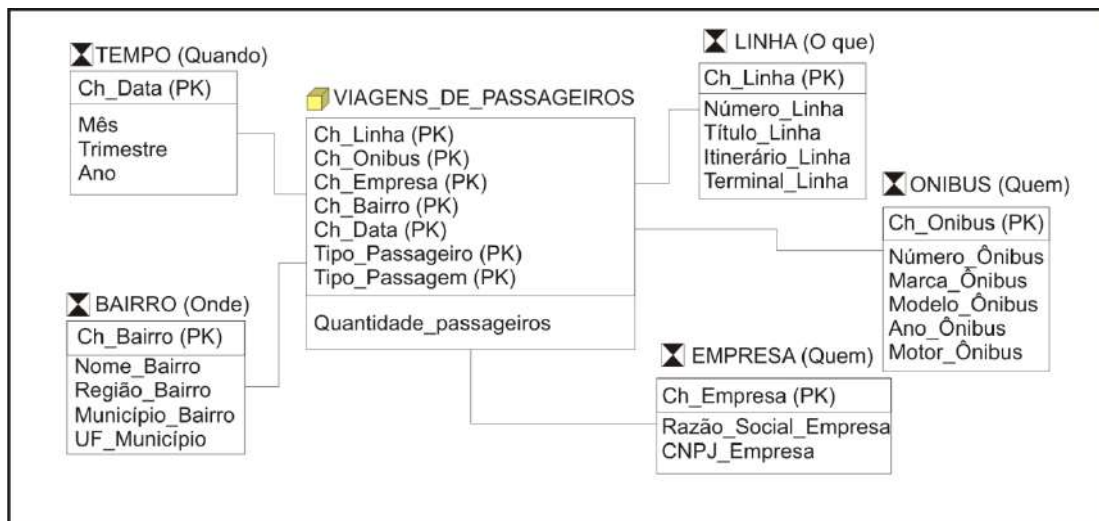


Figura 38 - Modelo dimensional final

O modelo físico do DW/DM construído aqui nesse estudo de caso é encontrado no Apêndice A e o script SQL de criação do banco para o modelo físico é encontrado no Apêndice C. No Apêndice B é possível encontrar o script de criação do banco de dados do sistema transacional apresentado na figura 15.

4 CONCLUSÕES E TRABALHOS FUTUROS

4.1 Conclusões gerais

O estudo de caso apresentado mostra de maneira eficiente como identificar fatos, medidas e dimensões de um negócio para construir um modelo dimensional de um DW/DM. A construção de um modelo lógico foi apresentada neste trabalho, baseada nas necessidades apresentadas pelo cliente e pela análise do modelo ER de um sistema legado desse cliente. Foi possível observar a importância da análise da granularidade desejada para os dados, observar a importância de analisar as hierarquias apresentadas pelas dimensões, podendo assim definir se há realmente a necessidade de normalizar alguma dimensão ou apenas utilizar essas hierarquias como atributos da dimensão. Foi possível identificar algumas dimensões “fracas”, dimensões pouco descritivas e com apenas um atributo, e analisar se estas dimensões devem ser transformadas em tabelas ou apenas utilizar do artifício de dimensão mascarada, que sugere transformá-las em atributos na tabela de fatos.

Os demais conceitos apresentados na fundamentação teórica, certamente, serão trabalhados em novos estudos de caso e em trabalhos futuros.

4.2 Trabalhos futuros

Como trabalhos futuros, esperam-se montar um grupo científico para estudos sobre *Business Intelligence* (BI) e levar adiante um projeto para soluções de BI para empresas de transporte coletivo urbano, visando não apenas a área de viagens de passageiros, mas também outras áreas de uma empresa, ou um grupo de empresas de transporte coletivo, bem como objetivos de instituições governamentais para acompanhamento, melhoria e evolução tecnológica dos sistemas de transportes coletivos urbanos.

A idéia para viabilizar o grupo científico é pesquisar e estudar ferramentas de BI *open source* (código fonte livre), como as do projeto *Pentaho* [PENTAHO, 2009], para desenvolver projetos inovadores, utilizando ferramentas gratuitas e que ainda podem ser alteradas para uma melhor utilização.

Além do projeto para soluções em transportes coletivos o grupo científico poderá desenvolver novos projetos e procurar ficar atualizado sobre as tendências e evolução do *Business Intelligence* para difundir o conhecimento nesta instituição de ensino e na comunidade acadêmica em geral por meio da realização de *workshops* sobre BI e trabalhos realizados, e por meio da escrita e publicação de artigos.

5 REFERÊNCIAS

ANDERSON, Kristin; KERR, Carol. **Customer Relationship Management**. 1ª ed. McGraw-Hill. 2001. 168 p.

ARAÚJO, Erika M. T. BATISTA, Mônica de L. S. MAGALHÃES, Teresinha M. de. **OLAP: Características, Arquitetura e Ferramentas**. Juiz de Fora, MG: Jornal Eletrônico, Instituto Vianna Júnior. 2007.

BARBIERI, Carlos. **Business Intelligence: Modelagem e Tecnologia**. Rio de Janeiro: Axcel Books. 2001.

BERNERS-LEE, T.; FIELDING, R.; FRYSTYK, H. **Hypertext Transfer Protocol -- HTTP/1.0**. Disponível em: <http://www.rfc-editor.org/cgi-bin/rfcdoctype.pl?loc=RFC&letsgo=1945&type=ftp&file_format=txt>. 1996. Acesso em: 2009.

BONOMO, Peeter. **Arquitetura de Data Warehouse**. Disponível em: <<http://imasters.uol.com.br/artigo/11417>>. 2009. Acesso em: 2009.

BRODIE, M. STONEBRAKER, M. **Migrating Legacy Systems: Gateways, Interfaces and the Incremental Approach**. Morgan Kaufmann Publishers, Inc. 1995.

COGNOS. **Business Intelligence and Performance Management Software Solutions from IBM**. Disponível em: <<http://www.cognos.com/>>. Acesso em: 2009.

DATE, C. J. **Introdução a Sistemas de Banco de Dados**. 8ed. São Paulo: Campus. 2004.

FELBER, Edmilson J. W. **Proposta de uma ferramenta OLAP em um Data Mart Comercial: Uma aplicação prática na indústria calçadista**. Trabalho de Conclusão de Curso. Centro Universitário FEEVALE. Novo Hamburgo. 2005.

INMON, W. H. **Como construir o Data Warehouse**. Rio de Janeiro: Editora Campus, 1997.

KIMBALL, Ralph. **Data Warehouse Toolkit: Técnicas para construção de data warehouses dimensionais**. São Paulo: Makron Books. 1998.

KIMBALL, Ralph; ROSS, Margy. **The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling**. 2ed. New York: John Wiley & Sons. Inc. 2002.

LAUDON, Kenneth C. **Sistemas de Informações Gerenciais: Administrando a empresa digital**. São Paulo: pp. p61.

MACHADO, Felipe N. R. **Tecnologia e Projeto de Data Warehouse: Uma Visão Multidimensional**. 3ed. São Paulo: Érica, 2007.

MICROSOFT. **Microsoft BI**. Disponível em: <<http://www.microsoft.com/brasil/servidores/bi/>>. Acesso em: 2009.

MICROSTRATEGY. **Business Intelligence Software Solutions**. Disponível em: <<http://www.microstrategy.com.br/>>. Acesso em: 2009.

MOSS, Larissa T.; ATRE, Shaku. **Business Intelligence Roadmap: The Complete Project Lifecycle for Decision-Support Applications**. Boston: Pearson Education, Inc. 2003.

OLIVEIRA, Celso H. P. de. **SQL Curso Prático**. São Paulo: Editora Novatec. 2002.

ORACLE. **Oracle Enterprise Performance Management e Business Intelligence**. Disponível em: <http://www.oracle.com/global/br/solutions/business_intelligence/index.html>. Acesso em: 2009.

PADOVOZE, Clóvis Luís. **Sistemas de informações contábeis: fundamentos e análise**. São Paulo: pp. p68.

PENTAHO. **Pentaho - Pay less for your Business Intelligence with the commercial alternative for BI**. Disponível em: <<http://www.pentaho.com/>>. Acesso em: 2009.

QLIKVIEW. **QlikView**. Disponível em: <<http://www.qlikview.com/>>. Acesso em: 2009.

QUEIROZ, Juliano. **SGBD: O que é?** Disponível em: <<http://espacoinfo.net/o-que-e-sgbd-bd-ii/>>. Acesso em: 2009.

SAP. **SAP BusinessObjects Intelligence Platform**. Disponível em: <<http://www.sap.com/solutions/sapbusinessobjects/large/intelligenceplatform/index.epx>>. Acesso em: 2009.

SILBERSCHARTZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. **Sistema de Banco de Dados**. 3ed. São Paulo: Pearson Makron Books. 1999.

SING, Hary. **Data Warehouse**. São Paulo: Makron Books, 2001.

SOMMERVILLE, Ian. **Engenharia de Software**. 8ª ed. São Paulo: Pearson Addison-Wesley. 2007.

VARGAS, Marcelo Feijó. **Construção de Data Warehouse para pequenas e médias empresas usando Software Livre**. Trabalho de Conclusão de Curso. Universidade do Planalto Catarinense. Lages. 2008.

WIKIPEDIA. **Joint Application Development - Wikipédia, a enciclopédia livre**. Disponível em: <http://pt.wikipedia.org/wiki/Joint_application_design>. Acesso em: 2009.

ZORZIN, Kênia Dias. **Implementação de um Data Warehouse para auxiliar nas tomadas de decisões do Hemocentro Coordenador de Palmas**. Trabalho de Conclusão de Curso. Centro Universitário Luterano de Palmas. Palmas. 2006.

A IMPLANTAÇÃO DO PROCESSO DE BUSINESS INTELLIGENCE-BI NA EMPRESA SAMARA CALÇADOS

Jacqueline Pessoa de Araújo
Fábio Nicácio de Medeiros

RESUMO

Com o advento da globalização, a competitividade entre as empresas no mundo dos negócios vem aumentando intensamente nos últimos anos. Logo nasceu a necessidade de uma nova visão empresarial para tomada de decisões a partir de análise de dados, o *Business Intelligence* (BI). O objetivo deste trabalho é apresentar um estudo de caso realizado na empresa Samara Calçados com intuito de demonstrar todos os passos de um processo BI e extrair informações para auxiliar o gerenciamento estratégico da empresa. Para tal, foi realizada uma pesquisa bibliográfica com o intuito de investigar os conceitos de BI, também realizado o processo *Extraction, Transformation and Load* (ETL) e por fim construídos relatórios com gráficos diversos, de forma simplista com intuito de facilitar a navegação dos gestores nas diversas dimensões e valores, utilizando os dados do *Data Warehouse* (DW) e ferramentas de *Online Analytical Processing* (OLAP) da *QlikView*.

Palavras-Chave: *Business Intelligence. Data Warehouse. QlikView.*

1. INTRODUÇÃO

Na era do conhecimento e da informação, as empresas começaram a se preocupar com maneiras de como se destacar no mercado de trabalho, buscando eficiência, melhor qualidade e menores custos nos processos. Esses fatores tornam-se os seus principais objetivos impulsionadas principalmente, pela crescente concorrência nos mercados globais, e eram com que várias empresas modificassem a forma de administrar seus processos. Alguns desses processos tiveram que ser repensados e modificados completamente, enquanto outros eram focados apenas nas partes que apresentavam deficiência, para que, só então, sua solução fosse procurada.

Os gestores e diretores passaram a entender que a área de tecnologia da informação (TI) poderia contribuir muito com as empresas, com a redução de pessoas na produção e/ou/escritórios, encurtando distâncias e diminuindo o tempo a ser gasto nas análises e em relatórios do sistema OLTP, que armazenam informações diárias.

Os dados estão em toda a parte, entretanto é preciso transformá-los em informações, e estas em conhecimento. A maioria das organizações não sofre com a falta de dados, mas sim com uma abundância de dados redundantes e inconsistentes, complexos de administrar com eficiência, cada vez mais árduos de acessar e difíceis de usar para fins de suporte à tomada de decisão. A informação é a chave para a vantagem competitiva no mundo dos negócios.

Com esta necessidade das empresas, surge o *Business Intelligence* (BI), que consiste na utilização de ferramentas tecnológicas para realizar extração, armazenamento e transformação dos dados contidos nos bancos de dados tradicionais

em informações que sirva para auxiliar na identificação de novas tendências, no conhecimento dos costumes dos clientes, fornecedores, concorrentes e o acesso às informações; em suma, o BI deve ser uma tecnologia de auxílio à tomada de decisões da alta gerência (PRIMAK, 2008).

Dessa forma foi necessário inovar para ganhar a concorrência. Com o *Business Intelligence*, à tomada de decisões passa a ser rápida, já que, para as análises antes feitas diante de relatórios de sistemas OLTP, passaram a ser gerados painéis de visualização para o usuário, os quais unem dados nos quais são comparadas as informações, transformadas em resultados.

Nessa pesquisa é realizado um estudo bibliográfico das principais tecnologias de BI, foi como também um estudo de caso para a implantação de BI na Empresa Samara Calçados.

1.1 JUSTIFICATIVA

O desenvolvimento deste trabalho foi baseado na necessidade de analisar informações da empresa Samara Calçados. O *Business Intelligence* (BI) é importante, pois tem a intenção de analisar um grande volume de dados de forma simplista, para auxiliar na tomada de decisões, mostrando que é possível investir em tecnologias para reduzir custos e, ainda, oferecer melhores serviços aos gestores da empresa, obtendo, com isso, soluções para os principais problemas da empresa.

1.2 OBJETIVO

1.2.1 Objetivos Gerais

Caracterizar a implantação de *Business Intelligence* (BI) e as etapas deste processo na Empresa Samara Calçados, localizada em João Pessoa-PB.

1.2.2 Objetivos Específicos

- ✓ Analisar a base de dados da empresa
- ✓ Realizar o processo de extração, transformação e carregamento dos dados
- ✓ Demonstrar como é desenvolvido o processo de BI
- ✓ Conhecer ferramentas o desenvolvimento
- ✓ Criar visões para análise de informações

1.3 ORGANIZAÇÃO DO TRABALHO

- ✓ No Capítulo 02 serão abordados os conceitos fundamentais sobre *Business Intelligence*, é descrito seu processo e seus ambientes, nos quais são desenvolvidas e demonstradas suas ferramentas.
- ✓ No capítulo 03 será abordado o estudo de caso de um projeto na empresa Samara Calçados, com a utilização do *software QlikView*.
- ✓ No capítulo 04 serão apresentados o resultado, as conclusões finais, as dificuldades encontradas durante o desenvolvimento da pesquisa e as sugestões propostas à elaboração de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

No mundo globalizado e competitivo de hoje, multiplicam-se as fontes de dados, tornando cada vez mais difícil o acesso às suas informações. Diante desta dificuldade em analisar dados, surgiu o *Business Intelligence* (BI), um conjunto de tecnologias ou ferramentas criado com a finalidade de auxiliar empresas a gerenciarem e transformarem seus dados em informações de valor. Por volta da década de 80, o BI passou a ser o responsável por acabar de vez com a era do “achismo”; logo, empresas passaram a ter informações reais sobre os processos de decisões (MACHADO, 2008).

Para ser eficiente e eficaz, o BI faz uso de ferramentas tecnológicas adequadas que permita organizar dados dispersos e fazer comparações na análise dos dados. A seguir serão abordados conceitos de *Business Intelligence*, as arquiteturas e características de *Data Warehouse*, o Modelo Dimensional, e a diferença para o Modelo Entidade Relacional (ER). Além de definições sobre *Data Mart*. O mesmo também aborda, as definições sobre OLAP, expondo de forma simplista as operações OLAP mais comuns e os tipos de ferramentas OLAP.

2.1 BUSINESS INTELLIGENCE

Primak (2008) define *Business Intelligence* como um processo inteligente de coleta, fazendo organização, análise, compartilhamento e monitorando dos dados inseridos em *Data Warehouse / Data Mart*, que serão apresentados nas seções seguintes, com a produção de informações para o suporte à tomada de decisões no ambiente de negócios, de forma a torná-lo mais eficiente.

As aplicações de BI vêm crescendo à medida que o mercado exige rapidez e diferencial competitivo. Elas permitem encontrar respostas em diferentes setores, e, com isso, podem tomar rumos diferentes e mostrar novas oportunidades, implicando nas direções dos segmentos, até nos *marketings* das empresas.

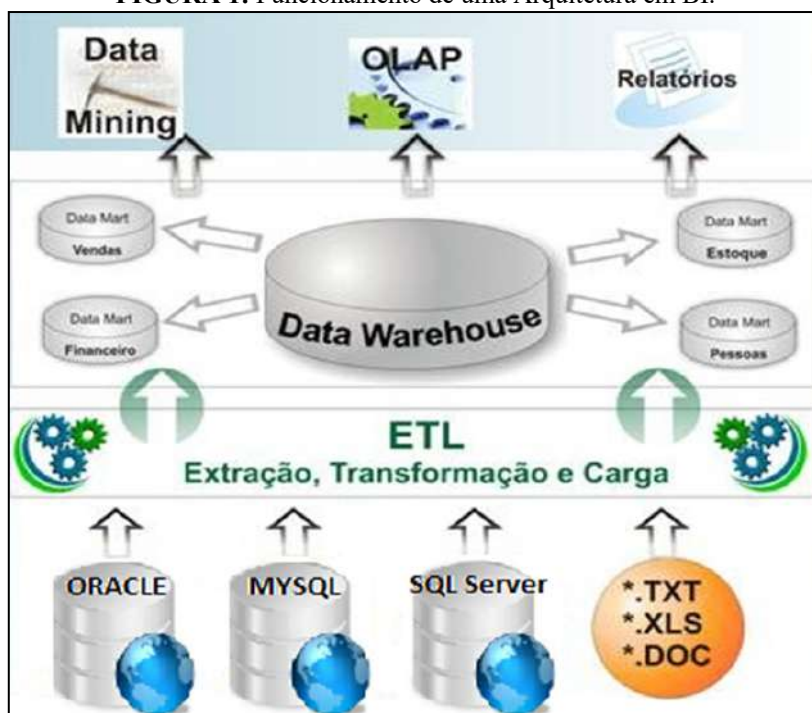
O *Business Intelligence* tem como objetivo principal a integração dos aplicativos e das tecnologias para extrair e analisar os dados corporativos de maneira simples, no formato correto e no tempo certo, para que a empresa possa tomar decisões melhores e mais rápidas, auxiliando os executivos em seus negócios. Esta ferramenta tem como capacidade transformar dados em informação, informação em conhecimento e, propiciar a descoberta de novos conhecimentos (KIMBALL, 2004).

As ferramentas de BI não possuem necessariamente um conjunto com as mesmas funcionalidades. É importante ressaltar que ferramentas específicas de BI podem prestar funcionalidades particulares. Algumas realizam apenas a função de extração de dados ou de análise, como também, fornecem uma suíte de serviços a exemplo da ETL (*Extract, Transform and Load*); outras, como a OLAP, realizam as análises (KIMBALL, 1998).

Conforme apresentado na Figura 1, a última camada possui várias bases de dados diferentes, nos quais é realizado o processo ETL, estudado na seção seguinte. Nela são disponibilizados meios e ferramentas para extrair, transformar e carregar os dados de diferentes fontes para uma única base, não volátil, chamada *Data Warehouse* (DW), que é uma tecnologia de armazenamento de dados organizados em um único ponto ou em subconjuntos denominados *Data Marts*. A organização dos dados sobre o modelo dimensional permite a interpretação dos mesmos de acordo com uma visão analítica, através de ferramentas OLAP (*On Line Analytical Processing*), abordada na seção seguinte. Tais ferramentas se caracterizam por permitir análises estratégicas dos processos de negócio que possam ser convertidas em tomadas de decisões e/ou permite essa interpretação sobre o prisma inferencial, através de técnicas de *Data Mining*,

caracterizadas pela detecção e análise de padrões para construção do conhecimento e em relatórios.

FIGURA 1: Funcionamento de uma Arquitetura em BI.



FONTE: <http://pedrofsi.blogspot.com/>

Nesta figura, é apresentada a arquitetura de um BI, a qual realiza processos de integração de fontes diferentes passando pela etapa de extração, transformação e carga em um único repositório de dados denominado de *Data Warehouse*. Após esta etapa os dados são entregues após terem sido organizados por meio de relatórios, OLAP (*On-line Analytical Processing*) ou mineração de dados, com o intuito de cooperar na tomada de decisões.

2.2 EXTRACT, TRANSFORM AND LOAD

É a fase mais complexa de um *Data Warehouse*, pois envolve a fase de movimentação de dados, cujas características são seus três passos: a extração, a transformação e, por fim, a carga dos dados.

Segundo *Wirthmann* (2003), a primeira parte do processo de ETL é a extração desses elementos dos sistemas originais. A maioria dos projetos de *Data Warehouse* consolidam dados extraídos de diferentes sistemas de origem. Cada sistema pode também utilizar um formato ou organização diferente. A extração converte de um determinado formato para a entrada no processamento da transformação.

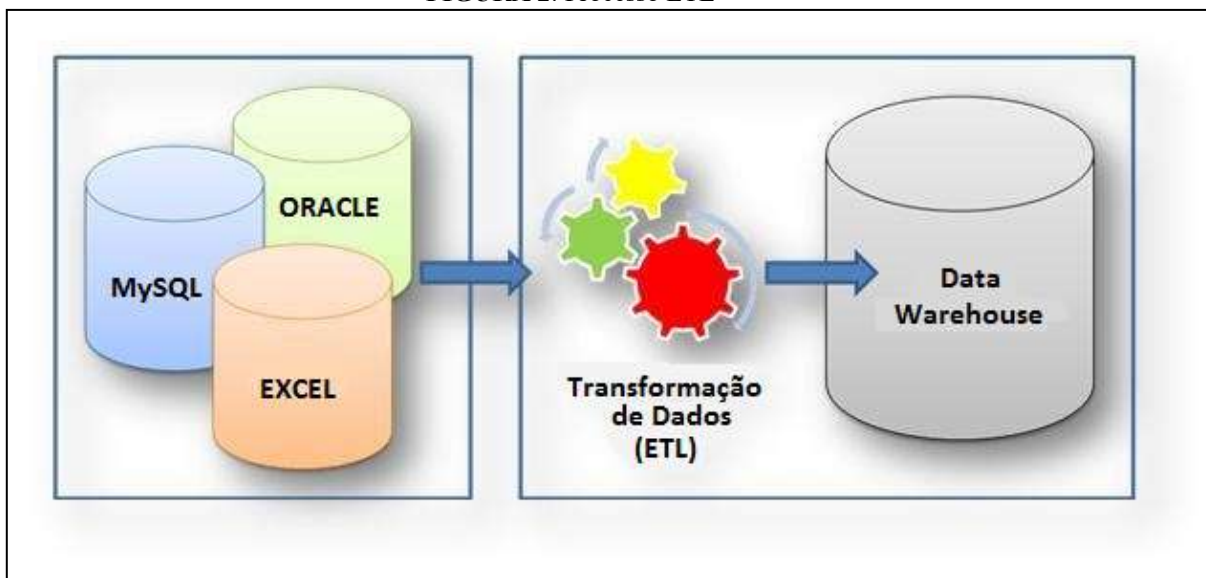
A segunda parte do processo consiste em transformar e limpar esses dados. Além da limpeza, é feita na maioria das vezes uma transformação, pois os dados provêm de vários sistemas, por isso, geralmente uma mesma informação tem diferentes formatos.

Por fim a terceira fase, a fase da carga de dados que carrega os dados no *Data Warehouse*. Alguns DW podem substituir as informações existentes semanalmente, com dados cumulativos e atualizados, ao passo que outro DW pode adicionar dados a cada hora. Sistemas mais complexos podem manter um histórico e uma pista de auditoria de todas as mudanças sofridas pelos dados.

Para Machado (2008), os processos de um DW são baseados na extração de dados, na organização, na integração, e no acesso aos dados para consultas.

A ferramenta ETL tem uma grande valia, principalmente se os sistemas transacionais são muitos, pois elas são uma poderosa fonte de geração de metadados, e que contribui muito para a produtividade da equipe, é importante identificar a ferramenta mais adequada para ser utilizada em um determinado caso, por existir um alto investimento, tanto a capacitação, quanto na aquisição. O fato verdadeiro é que os benefícios serão bastante vistosos e a produtividade aumentará consideravelmente.

FIGURA 2: Processo ETL



FONTE: <http://www.bligoo.com/explore/tag/datamart>

Nesta figura, o processo ETL, configura-se pelo agrupamento de diversos sistemas ou tabelas diferentes em um único modelo. Para que isso ocorra, é utilizado o processo de extração, o qual extrai informações de tabelas e sistemas diferentes, a seguir é feita a etapa de transformação dos dados, a fase mais crítica do processo, nela são unidas informações de uma mesma origem mais de forma diferente como, por exemplo, em um sistema onde o sexo é feminino e masculino, e em outro sistema onde o sexo está como F ou M, podem ser transformados para um único tipo, sendo disposto no repositório de dados conhecidos como *Data Warehouse*.

2.3 DATA WAREHOUSE

Segundo Kimball (2004), o *Data Warehouse* extrai e limpa dados de origem, em um armazenamento de dados dimensional e, em seguida, apoia e implementa consulta e análise para ajudar na tomada de decisão. O DW gerencia o fluxo de informações a partir dos bancos de dados corporativos e fontes de dados externas à empresa, e pode ser definido como um banco de dados especializado.

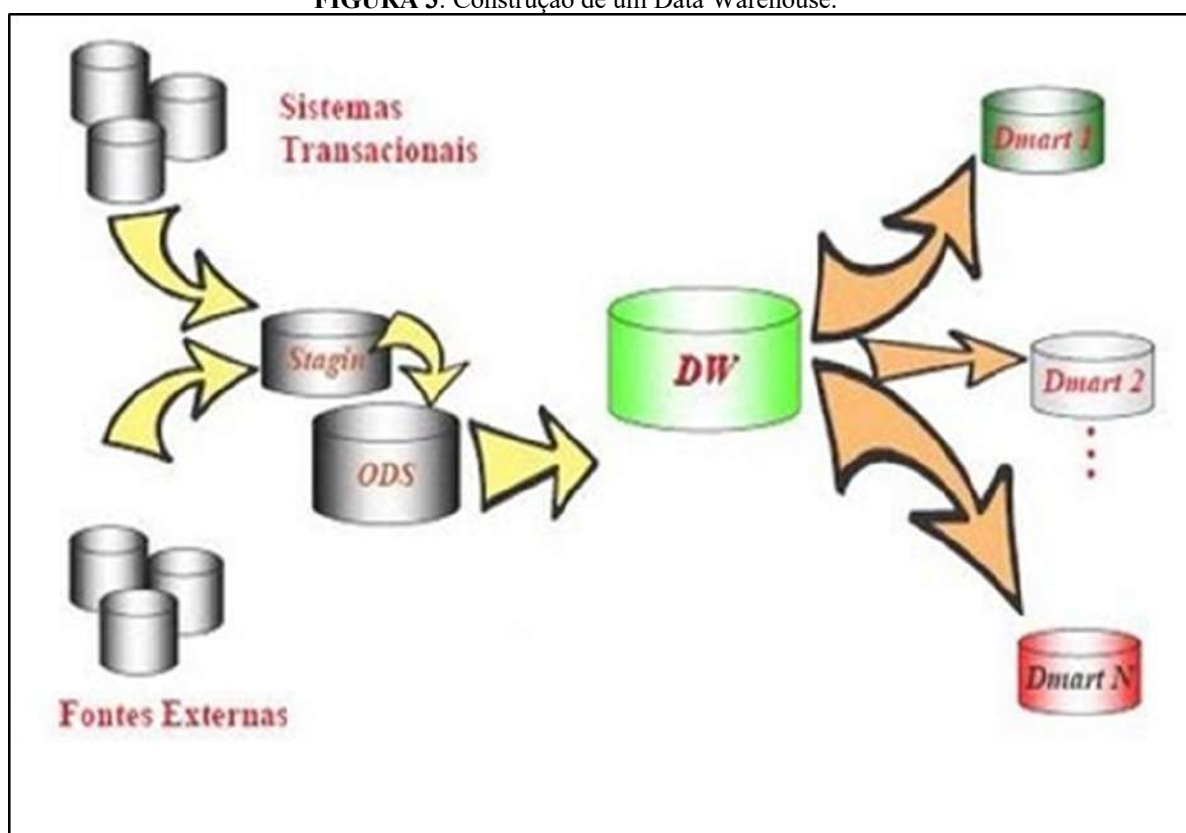
Os *Data Warehouse* são bem diferentes dos bancos de dados transacionais, quanto a aspectos da estrutura; na maioria das vezes, contém grandes quantidades de dados extraídos de múltiplas fontes, podendo incluir diversos bancos de dados e até mesmo arquivos adquiridos de sistemas e plataformas independentes. Já com relação à funcionalidade, fornece a possibilidade de analisar dados para apoiar as consultas de informações, sendo necessário o uso de uma ferramenta específica para esta etapa. No

desempenho há um ganho de eficiência em relação aos bancos de dados transacionais uma vez que o *Data Warehouse* permite consultas completas.

Os dados analíticos armazenados em um DW são destinados às necessidades da gerência no processo de tomada de decisões. Isto pode envolver consultas complexas que necessitam acessar um grande número de registros; por isso, é necessária a existência de muitos índices criados para acessar as informações da maneira mais rápida possível. Os DWs armazenam informações históricas de muitos anos e, por isso, devem ter uma grande capacidade de processamento e armazenamento dos dados.

O principal objetivo de um *Data Warehouse* é disponibilizar informações retiradas dos bancos de dados para apoio as decisões da empresa. Tal construção exige a transferência e transformação de dados existentes de uma base de dados corporativa, para controle diário de operações, em uma base de dados independente (MACHADO, 2008).

FIGURA 3: Construção de um Data Warehouse.



FONTE: <http://luiseduardotrovo.blogspot.com/2009/08/componentes-de-um-dw.html>

Nesta figura, é apresentado como é construído um *Data Warehouse*, com extrações de dados em um único repositório assim denominado. Após esta etapa, os dados estão organizados em *Data Marts* conhecidos como prateleiras.

2.3.1 Características do Data Warehouse

De acordo com Machado (2008), o DW tem como características ser orientado por assunto, integrado, não volátil, variável com o tempo.

A orientação por assunto é uma característica importante, pois toda a modelagem do DW é orientada a partir dos assuntos mais importantes da empresa. Os sistemas transacionais operacionais são orientados a processos, trabalham com os dados diários

das atividades que envolvem o processo de funcionamento da organização, dados que existem, basicamente, para fins de controle operacional e, no geral, não apoiam decisões de negócio.

A integração é a característica mais importante do *Data Warehouse*, pois seu objetivo é integrar dados no ambiente operacional, para as aplicações do DW. A integração é realizada visando padronizar os dados dos diversos sistemas em uma única representação, para serem transferidos para a base de dados única do *Data Warehouse*.

Em sistemas transacionais com características não-voláteis, os dados sofrem diversas alterações, como, por exemplo, a inclusão, a alteração e a exclusão de dados. No ambiente do *Data Warehouse* os dados, antes de serem carregados, são filtrados e limpos “gerando informações”. Após esta etapa, esses dados sofrem somente operações de consulta e exclusão, sem que possam ser alterados (BARBIERE, 2001).

Por fim, a variação em relação ao tempo consiste na manutenção de um histórico de dados em relação ao período de tempo maior que o dos sistemas comuns, de forma a não comprometerem o desempenho dos bancos transacionais OLTP. Ao analisarmos um dado de um DW, o mesmo sempre estará relacionado a um período determinado de tempo, pois terá uma chave de tempo que irá indicar o dia no qual esses dados foram extraídos.

2.3.2 Arquitetura de Data Warehouse

A arquitetura depende do local onde os *Data Warehouse* ou *Data Marts* estão residindo. As arquiteturas abordadas serão: arquitetura global (centralizada e descentralizada), arquitetura de *Data Marts* independentes, arquitetura de *Data Marts* integrados.

A escolha da arquitetura é uma decisão gerencial do projeto, e está normalmente baseada nos fatores relativos à infra-estrutura disponível, ao ambiente de negócios (parte da empresa), concomitantemente com o escopo de abrangência desejado, assim como a capacitação dos empregados da empresa e dos recursos disponibilizados ou projetados para investimento. (MACHADO, 2008, p.47)

Na arquitetura global, o projeto é feito com base nas necessidades da empresa como um todo e tem como objetivo construir um repositório de dados de suporte à decisão disponível para toda a organização. Existem dois caminhos na arquitetura global. São eles: a arquitetura global distribuída que é utilizada quando uma empresa possui diversos locais físicos e dados em diversas instalações, e arquitetura global centralizada, quando a empresa existe em um único local.

Os dados são extraídos de sistemas operacionais e de fontes de dados externos por processos de *batch* (dependente do passo anterior para sua continuação) em horários diferentes de operações. Eles são filtrados, pois os dados não-necessários são eliminados, e é realizada a transformação para a qualidade e necessidade dos requisitos do projeto. Por fim, são carregados no *Data Warehouse*, para acesso dos usuários finais.

A grande vantagem desta arquitetura é que os usuários podem utilizar visões corporativas dos dados, porém, há desvantagens, como o consumo de tempo, já que, o desenvolvimento e administração de um ambiente sobre esta arquitetura é muito grande, assim como o custo de sua implementação.

Já a arquitetura de *Data Marts* independente não é corporativa. Seu principal foco são consultas que encantam os usuários. Não existe conexão entre os DMs, os quais são isolados por departamento ou por grupos específicos de usuários. Esse tipo de arquitetura geralmente resulta em uma implementação bastante rápida, devido ao

escopo reduzido e isolado, porém fica claro que a capacidade de decisão através de dados corporativos é limitada.

Na arquitetura de *Data Marts* integrados, os dados são distribuídos em departamentos ou áreas de negócios; na arquitetura de *Data Marts* independentes, são interconectados, integrados e acessíveis por outros departamentos ou áreas não são mais isolados; portanto, esta arquitetura provê um aumento na capacidade de visão corporativa e uma maior qualidade das informações, porém aumenta o nível de complexidade dos requisitos, aumenta a necessidade de controle e de administração, e, certamente, aumenta o envolvimento do setor de tecnologia da informação da empresa.

2.3.3 Tipos de Implementação

As implementações abordadas a seguir são as mais utilizadas, como *top-down*, *bottom-up* ou a combinação das duas. A opção de escolha é influenciada por fatores da infraestrutura de tecnologia da informação, a arquitetura escolhida, os recursos disponíveis e a necessidade de acesso corporativo ou não.

2.3.3.1 Top-down

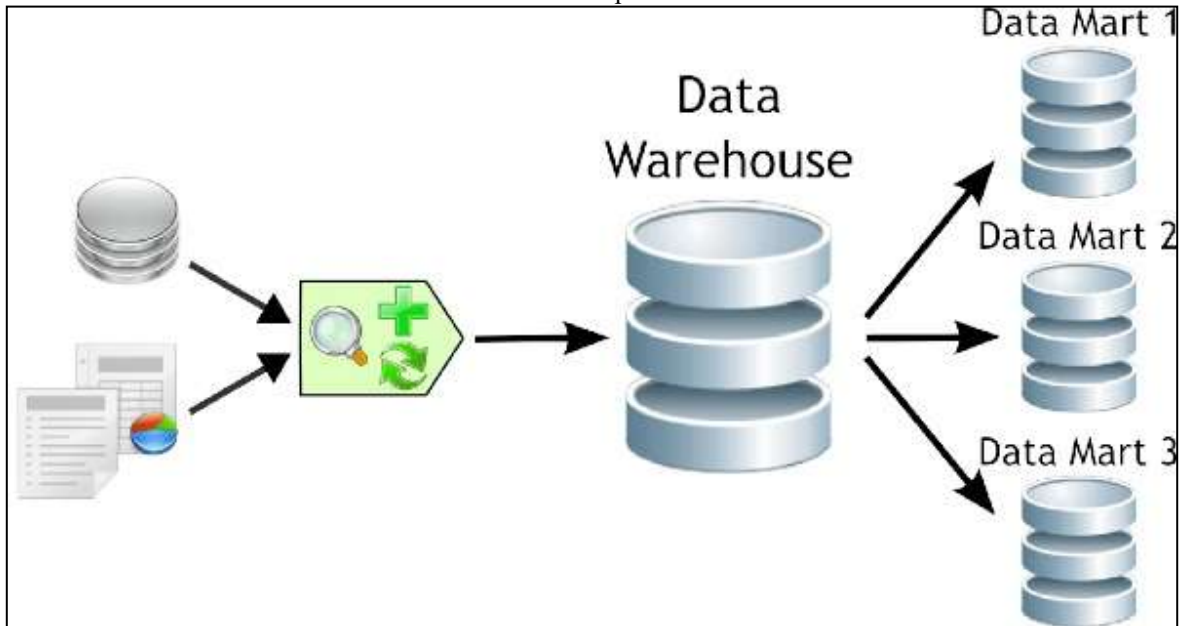
Na abordagem *Top-Down* centrada nos dados, Inmon (2005) define a organização inicial do projeto, ameniza a excessiva dependência dos dados presente na proposta inicial, Com âmbito, objetivos, áreas de assunto, abordagem e arquitetura, e em paralelo tem três fases que convergem para a fase do desenho físico: a definição da infraestrutura técnica, o desenho preliminar e a modelagem dos dados. As fases são o processo iterativo de desenvolvimento do DW com um fluxo em espiral.

A implementação *top-down* requer maior planejamento e trabalho de definições conceituais de tecnologia antes de se iniciar um projeto de *Data Warehouse*. Devem ser tomadas decisões sobre quais fontes de dados devem ser utilizadas, a segurança, a estrutura de dados, as qualidades de dados a serem considerados, os padrões de dados e diversos modelos de dados transacionais atuais que devem estar completos antes de se iniciar a implementação.

Segundo Machado (2008), as vantagens dessa abordagem são: heranças de arquitetura, já que os DM são originados de um DW e estes utilizam a arquitetura e os dados deste *Data Warehouse*, o que permite uma facilidade de manutenção; visão de empreendimento, pois o DW concentra toda a empresa e seus negócios; controle e centralização de regras, isto é, um único conjunto de aplicações para extração, transformação e integração dos dados, além de processos centralizados de monitoração e manutenção.

Como desvantagem, essa abordagem apresenta uma implementação muito longa, o que significa que esse DW leva muito tempo para entrar em produção, e a consequência disso é o alto risco de investimento, além da demora do retorno sobre esse investimento. É necessária uma equipe altamente capacitada para verificar as informações e consultas que garantam à empresa a possibilidade de sobreviver e prosperar nas mudanças. Por fim, uma demora no projeto e a falta de retorno, podem gerar expectativas no usuário. A arquitetura será detalhada abaixo.

FIGURA 4: Top-down



FONTE: http://www.dataprix.net/files/uploads/250image/HEFESTO%20v2_0/data%20mart%20-%20top%20down.png

Esta figura demonstra o processo *top-down*, no qual os dados são carregados nos *Data Warehouse* através do processo ETL. Depois eles são alimentados em *Data Marts* distintos, cada um com os dados referentes ao assunto ou departamento.

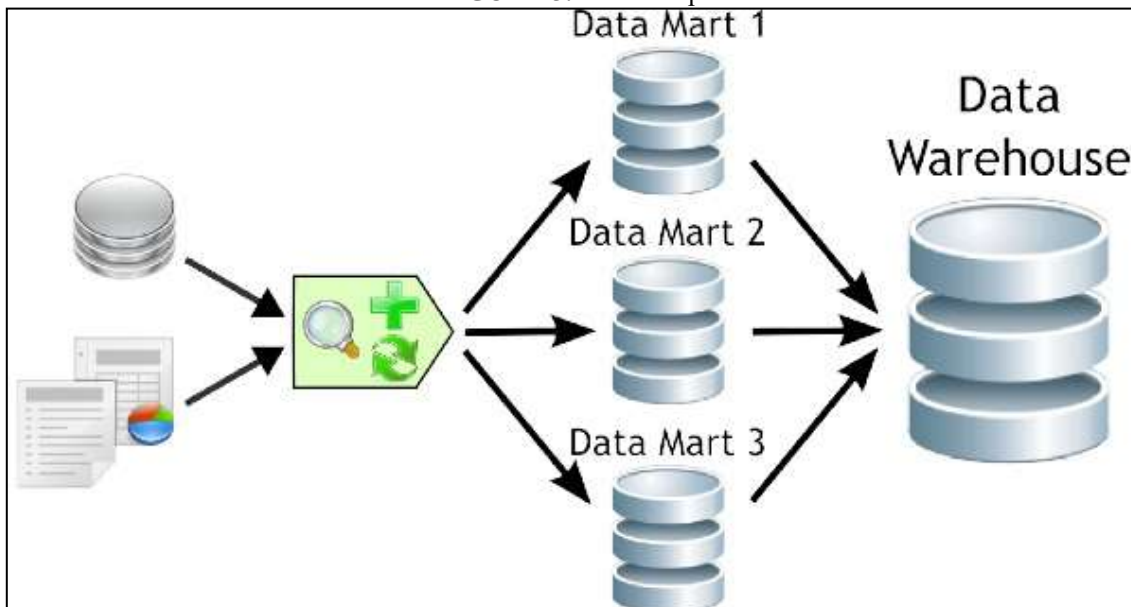
2.3.3.2 *Bottom-up*

Nessa implementação, o planejamento e o desenho dos *Data Marts* podem ser realizados sem a necessidade de definir a infraestrutura corporativa para *Data Warehouse* na empresa. O propósito do *bottom-up* é a construção de uma DW incremental a partir do desenvolvimento de DM independentes.

As vantagens dessa implementação são: a construção dos DM é altamente direcionada, permitindo um rápido desenvolvimento; a arquitetura em DM com incremento demonstra um retorno rápido. A elaboração de DMs incrementais permite que os negócios sejam enfocados inicialmente, evitando gastos em áreas não-essenciais; e a estratégia de DMs incrementais obriga a entrega de informações passo a passo, permitindo à equipe crescer e aprender, o que reduz os custos.

Já as desvantagens da implementação *bottom-up* consistem na criação de DMs independentes. Logo estes transformam-se em DM legados, que dificultam e inviabilizam futuras integrações. É necessário que se mantenha um rígido controle de negócios, e isso requer mais trabalho, ao extrair e combinar as fontes individuais do que utilizar um DW. O desenvolvimento em paralelo leva a uma rígida administração, tentando coordenar os esforços e recursos de várias equipes. Por último, os usuários finais dos DMs estão felizes e querendo mais informações para seus DMs, enquanto que outros usuários aguardam o incremento de seus DMs. A figura a seguir mostra esta implementação:

FIGURA 5: Bottom-Up



FONTE: http://www.dataprix.net/files/uploads/250image/HEFESTO%20v2_0/data%20mart%20-%20bottom%20up.png

Esta figura demonstra o processo *bottom-up*, no qual os dados são carregados nos *Data Marts*, através do processo ETL, em diversos DM diretamente, depois alimentados no *Data Warehouse*.

2.4 GRANULARIDADE

A granularidade dos dados é um dos fatores mais importantes a serem tratados no *Data Warehouse*, pois afeta profundamente o volume de dados e os tipos de consultas a serem realizadas.

Granularidade refere-se ao nível de detalhamento disponível nos dados. A figura a seguir mostra o baixo e alto nível de granularidade dos dados.

FIGURA 6: Alto e Baixo nível de granularidade dos dados



FONTE: Machado (2007)

Quanto menor a granularidade, mais detalhes os dados possuem, e quanto menor forem esses detalhes, maior sua granularidade.

2.5 MODELAGEM DE DADOS

A modelagem de dados é uma técnica usada para a especificação das regras de negócios e as estruturas de dados de um banco de dados. São consideradas duas técnicas diferentes de modelagem de dados: a modelagem ER e a modelagem multidimensional.

2.5.1 Modelo entidade relacionamento

De acordo com Machado (2008), o conceito de modelagem entidade relacional (ER) é aquele equivalente à metodologia antiga mais usada no mundo de banco de dados, para análise e projeto de sistema de informação.

O diagrama ER utiliza três símbolos gráficos:

- ✓ Entidades: Uma entidade é um objeto que existe no mundo real (concreto ou abstrato), com uma identificação distinta e com um significado próprio. Existe independentemente do fato de esta ligada a outros objetos do banco de dados. Pode ser definida como, por exemplo: uma pessoa, um lugar, um evento de interesse da empresa.
- ✓ Relacionamento: Um relacionamento é representado por linhas entre as entidades, em um diagrama ER. Esse relacionamento é definido em termos de cardinalidade. É o máximo número de instâncias de uma entidade que está relacionado com uma simples instância de outra entidade e vice-versa.
- ✓ Atributos: Atributo é o elemento de dados que contém as informações e características que descreve uma entidade. Um nome de atributo deve ser único em uma entidade e deve ser semântico suficiente para descrevê-lo. Com a comparação dos valores dos atributos nas duas entidades, se estabelece um relacionamento entre as suas ocorrências.

2.5.2 Modelagem Dimensional

Modelagem dimensional é um nome novo para uma técnica antiga usada para criar banco de dados simples e compreensíveis (KIMBALL, 1998). Modelos multidimensionais proporcionam uma estrutura de sistemas de informação que permitem a uma empresa dispor de acesso muito flexível a dados, fatie e agrupe dados de qualquer número de maneira explore dinamicamente o relacionamento entre dados resumidos e detalhados (INMON, 1997). Esses sistemas além de oferecerem flexibilidade em relação às necessidades dos usuários fornecem um alto nível de controle.

É um modelo otimizado para consultas, voltado para o entendimento do negócio e que dá flexibilidade para o usuário final. Como analogia, podemos pensar em “cubos de dados” que representam as necessidades de informação, também chamado de “*Star Schema*” ou “Modelo Estrela”, pois há uma tabela “principal” ao centro, rodeada de tabelas auxiliares que seriam as pontas da estrela. Portanto, é um modelo assimétrico. Denominamos a tabela central como Tabela de Fatos e as demais de Tabelas de Dimensão.

Uma modelagem multidimensional contém as mesmas informações que um modelo normalizado, mas empacota os dados em um formato cujos objetivos de *design* são a capacidade de ser compreendido pelo usuário, o desempenho de consulta e a flexibilidade de adaptação a mudanças. As estruturas normalizadas não devem abranger consultas de usuário porque elas prejudicam a capacidade de compreensão e o desempenho (Kimball, 2004).

Segundo Machado (2008), a modelagem multidimensional é uma técnica de concepção e visualização de um modelo de dados e de um conjunto de medidas que descrevem aspectos comuns de negócios.

Ela permite visualizar os dados de uma organização, fornecendo e analisando informações desses dados para a tomada de decisões.

Esse modelo é formado por três elementos básicos:

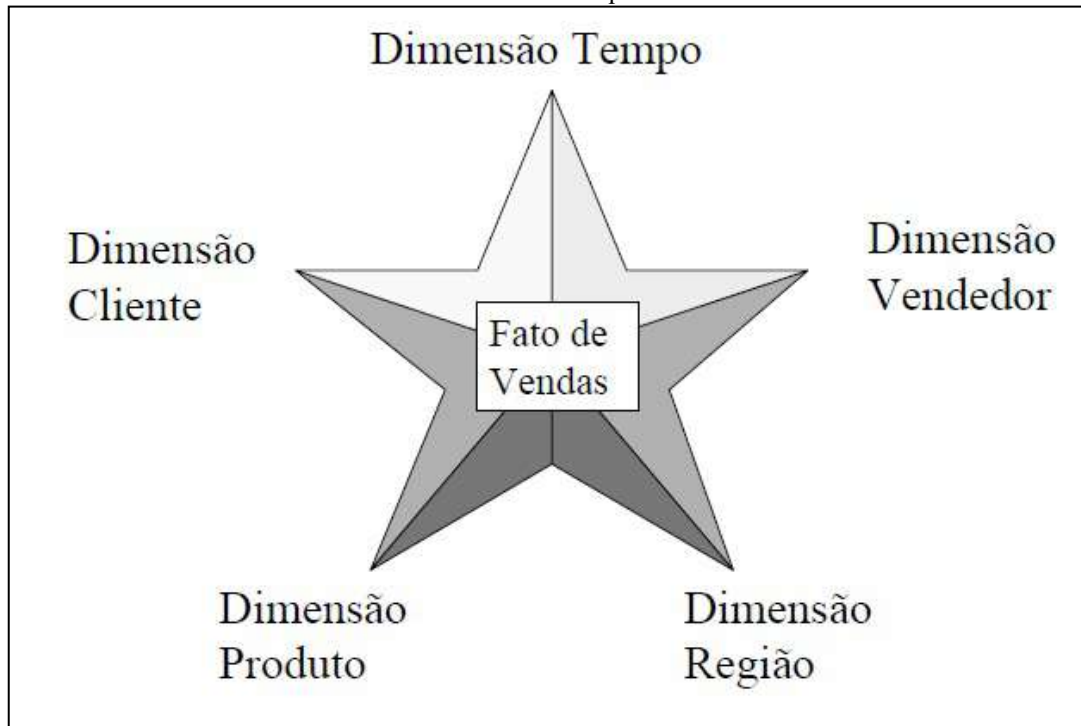
- ✓ Fatos: Fatos são os dados agrupados, possuindo valores de cada medida para cada combinação da dimensão existente. O tamanho da tabela que contém os fatos merece atenção preferencial do analista. A característica básica de um fato é que ele é representado por valores numéricos e implementado em tabelas de fatos (*fact table*);
- ✓ Dimensões: Estabelece a organização dos dados, determinando possíveis consultas. Representa uma perspectiva de análise de usuários de sistema de suporte a decisão (SSD). As dimensões, na maioria dos casos, não possuem atributos numéricos, pois são somente descritivas e classificatórias dos elementos que participam de um fato;
- ✓ Medidas: De acordo com Machado (2008), medidas são os atributos numéricos que representam um fato. Uma medida é determinada pela combinação das dimensões que interagem com um fato.

O objetivo dessa modelagem é fornecer a capacidade de visualizar os dados de uma organização, permite obter e analisar informações desses dados para a tomada de decisões.

2.5.2.1 STAR SCHEMA

O *star schema* (esquema estrela) é um modelo sobre uma estrutura básica em formato estrelar, simples e eficiente, caracterizada por ter uma única tabela de fatos e chaves simples nas tabelas de dimensões. As vantagens desse modelo são a eficiência, dada pelo pequeno número de junções pela pesquisa e chaves simples; e a facilidade de definir hierarquias. As desvantagens são o tamanho e a falta de normalização das tabelas de dimensão. Para Machado (2008), o modelo estrela é uma estrutura básica de um modelo de dados multidimensional.

FIGURA 7: Modelo Esquema Estrela



FONTE: http://www.cin.ufpe.br/~ejvm/OLAP/Datawarehouse_9pp.pdf

Na figura 7, a sua estrutura possui uma entidade central denominada fato (*fact table*) e um conjunto de entidades menores denominadas dimensões (*dimension tables*), em torno da entidade central, por isso o nome estrela pela sua estrutura. Esta tabela fato se relaciona com as outras dimensões.

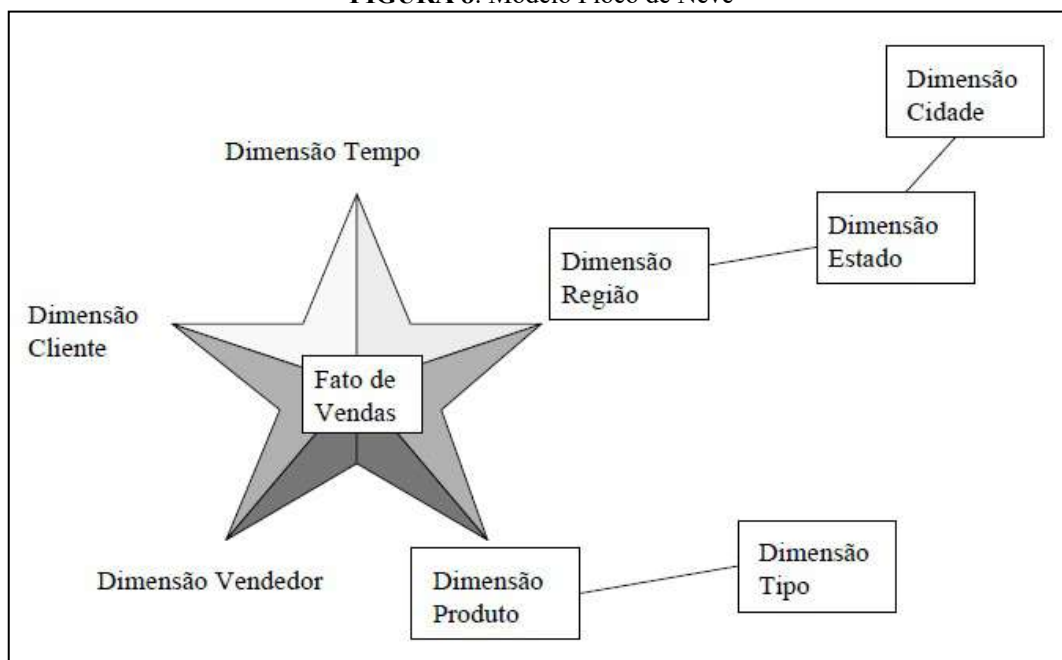
Vale ressaltar que este tipo de esquema não está restrito a apenas uma tabela fato, ou seja, é possível que haja mais de uma tabela fato se relacionando através de dimensões, porém é mais comum visualizar o esquema separado por fato devido a sua baixa complexidade na visualização.

2.5.2.2 SNOWFLAKE

O modelo *snowflake* (flocos de neve) começa com a identificação de fatos e dimensões, logo após serem elaborado os documentos de requisitos. É o resultado da decomposição de uma ou mais dimensões que possuem hierarquias entre seus membros. (Machado, 2008).

Com a aplicação da terceira forma normal nas entidades dimensões, sendo um modelo normalizado evita redundância, não é o objetivo dos DW, e consultas nesse modelo são mais lentas, visto que são necessárias mais junções na consulta de fatos de uma dimensão normalizada.

FIGURA 8: Modelo Floco de Neve



FONTE: http://www.cin.ufpe.br/~ejvm/OLAP/Datawarehouse_9pp.pdf

Na figura 8, a estrutura representada possui uma entidade central denominada fato (*fact table*) e um conjunto de entidades menores denominadas dimensões (*dimension tables*), ligando a estas dimensões novas dimensões secundárias que podem ser ligadas a outras dimensões terciárias, e enfim, várias outras em torno da entidade central, por isso, o nome floco de neve. Esta tabela fato se relaciona com as demais dimensões, sendo que, no modelo floco de neve, as dimensões secundárias e terciárias não se relacionam com a tabela fato, mas apenas com as com que estão ligadas.

2.6 DATA MART

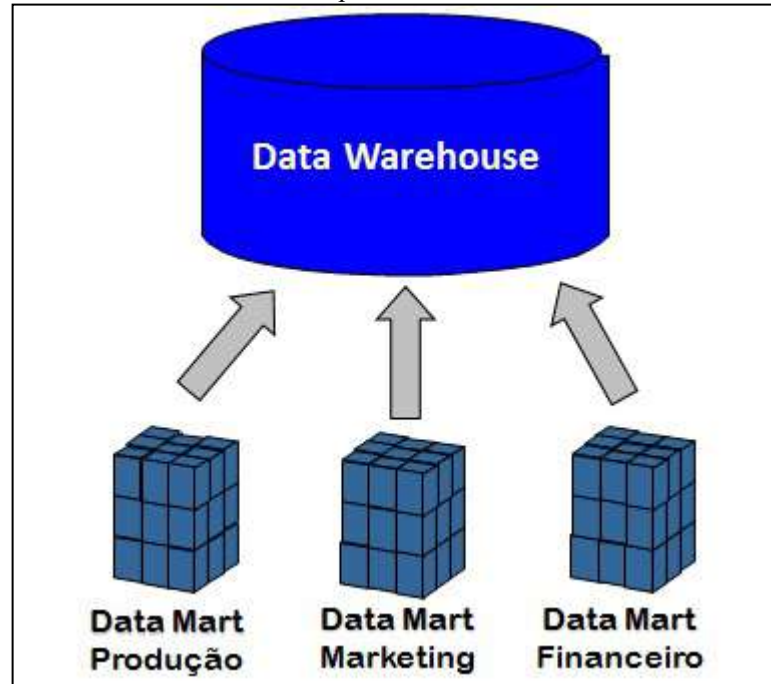
Para Kimball (2005), um *Data Mart* é um subconjunto de dados do *Data Warehouse* destinado a suportar as necessidades específicas de uma determinada unidade de negócios. Inmon (2005) afirma que os *Data Marts* dependem diretamente do *Data Warehouse*, podem conter subconjuntos, dados agregados ou atômicos, fornece uma visão departamental, tem a possibilidade de ficar numa plataforma diferente do DW.

Um *Data Warehouse* é a união de seus *Data Marts*. O DM é um conjunto de tabelas dimensionais que apoiam um processo de negócio. Um DM é um DW reduzido que fornece suporte à decisão de um pequeno grupo de pessoas.

Um *Data Mart* é uma porção física ou lógica de um *Data Warehouse*, para atender a uma área específica da empresa. Muitas vezes, DM são criados de forma a oferecer simplicidade, menor custo e agilidade ao processo de construção e manutenção do *Data Warehouse* (FELBER, 2005).

As vantagens de um *Data Mart* é o retorno rápido, que garante uma maior contribuição do cliente, capaz de avaliar os benefícios extraídos do investimento. A diferença principal entre esses sistemas é que o DM trata do problema do departamento local e o *Data Warehouse* trata da instituição como um todo.

FIGURA 9: Repositório Data Mart



FONTE: Autor Próprio

Esta figura demonstra um único repositório de dados denominado *Data Warehouse*, e suas divisões, que podem ser entendidas como departamentos de uma empresa.

2.7 FERRAMENTAS OLAP

Segundo Machado (2008), OLAP é uma ferramenta de *Business Inteligente* utilizada para apoiar as empresas na análise de suas informações, visando atingir apoio nas tomadas de decisões.

O termo OLAP refere-se a um conjunto de ferramentas voltadas para o acesso e a análise de dados, com o objetivo final de transformar dados em informações capazes de dar suporte a decisões gerenciais de forma flexível ao usuário e em tempo hábil. A ferramenta OLAP é capaz de efetuar cálculos complexos, como previsões, percentuais de crescimento e médias diversas, considerando-se a variável tempo. É uma ferramenta muito importante no contexto gerencial, ajudando a analisar, de forma mais eficiente, a quantidade de dados crescente armazenada pelas organizações, tornando as informações úteis (THOMSEN, 2002).

No OLAP, as informações são armazenadas em cubos multidimensionais que gravam valores quantitativos e medidas, permitindo a visualização através de diversos ângulos. Estas medidas são organizadas em categorias descritivas chamadas de dimensões, e formam, assim, a estrutura do cubo.

As ferramentas são as aplicações às quais os usuários finais têm acesso para a extração de dados de suas próprias bases e, a partir deles, gerar relatórios capazes de responder às suas questões gerenciais.

2.7.1 Características de OLAP

A funcionalidade de uma ferramenta OLAP é caracterizada pela análise multidimensional dinâmica dos dados, apoiando o usuário final nas suas consultas. As principais operações de uma ferramenta são (LEME FILHO, 2004):

- ✓ *Drill Across*: significa o salto dado sobre um nível intermediário, dentro de uma mesma dimensão, ocorre quando o usuário pula um nível intermediário dentro de uma mesma dimensão. Por exemplo, a dimensão tempo é composta por ano, semestre, trimestre, mês e dia. A operação *Drill Across* é executada quando o usuário passa de ano diretamente para trimestre ou mês;
- ✓ *Drill Down*: ocorre quando o usuário aumenta o nível de detalhe da informação, diminuindo a granularidade. A granularidade determina os tipos de consultas que podem ser feitas no DW. Ela influencia diretamente na velocidade do acesso às informações e no volume de dados armazenados;
- ✓ *Drill Up*: é o contrário do *Drill Down*, ocorre quando o usuário aumenta a granularidade, diminuindo o nível de detalhamento da informação;
- ✓ *Drill Throught*: é caracterizado pela mudança de dimensão. Assim, a análise da informação passa a ser efetuada sobre a outra dimensão. Pode analisar uma dimensão tempo para que, no próximo passo, se analise a informação por região;
- ✓ *Slice and Dice*: são operações para realizar a navegação por meio dos dados na visualização de um cubo. Consiste em fatiar os dados, permitindo fixar um determinado valor de uma dimensão e objetivando diminuir o escopo. É uma das principais características de uma ferramenta OLAP. Como a ferramenta OLAP recupera o microcubo, surgiu a necessidade de criar um módulo, que se convencionou de *Slice and Dice*, responsável por trabalhar esta informação. Ele serve para modificar a posição de uma informação, trocar linhas por colunas, de maneira a facilitar a compreensão dos usuários e girar o cubo sempre que houver necessidade;
- ✓ *Pivot* é a mudança ou troca do ângulo de visão dos dados. Consiste em trocar linhas por colunas, em uma tabela, ou trocar as dimensões de um gráfico.

2.7.2 Arquiteturas OLAP

Uma arquitetura OLAP guarda os dados em memória, deixando as consultas e informações dos bancos de dados mais rápidas.

De acordo com [LINS NETO, 2010 *apud* GOLFARELLI; RIZZI, 2007], as arquiteturas OLAP podem ser implementadas de diversas formas:

- ✓ MOLAP (*Multidimensional On Line Analytical Processing*);
- ✓ ROLAP (*Relational On Line Processing*);
- ✓ HOLAP (*Hybrid On Line Analytical Processing*);
- ✓ DOLAP (*Desktop On Line Analytical Processing*);
- ✓ WOLAP (*Web On Line Analytical Processing*).

2.7.2.1 MOLAP (MULTIDIMENSIONAL ON LINE ANALYTICAL PROCESSING);

A arquitetura MOLAP (*Multidimensional On Line Analytical Processing*) acessa os dados diretamente no banco de dados multidimensional, ou seja, é o usuário que faz

o trabalho. Através desta ferramenta, ele monta e manipula diretamente os dados do cubo diretamente do servidor.

Segundo Machado (2008), o banco de dados é acessado através de cubos e hipercubos. O termo “cubo de dados” é apenas uma aproximação da forma como os dados estão organizados, e não a real expressão de uma realidade. Os cubos de dados estão organizados de forma a agregarem valores em uma tabela, denominada tabela de fatos, com base em dimensões selecionadas para fazer parte do cubo.

Na arquitetura MOLAP, os dados de um banco multidimensional são armazenados em um espaço menor que o utilizado para armazenar os mesmos dados em um banco de dados relacional. No banco de dados multidimensional, os dados são mantidos em estruturas de dados do tipo *array*, de maneira que podem fornecer um melhor desempenho ao acessá-los. Além de ser uma arquitetura rápida, há outra vantagem: o rico e complexo conjunto de funções de análises presentes nos bancos de dados multidimensionais (Carvalho, 2004).

Uma de suas desvantagens é a possibilidade de os dados serem esparsos, ocorrendo a chamada explosão de armazenamento de dados, ou seja, um imenso banco de dados multidimensional contém poucos dados armazenados. Outra desvantagem dessa ferramenta está relacionada ao fato de os bancos multidimensionais serem sistemas proprietários que não seguem padrões, ou seja, cada desenvolvedor cria a sua própria estrutura para o banco e as próprias ferramentas de suporte.

2.7.2.2 ROLAP (RELATIONAL ON LINE PROCESSING)

Segundo Machado (2008), ROLAP é uma arquitetura que acessa banco de dados relacionais. Na arquitetura ROLAP, a consulta é enviada ao servidor de banco de dados relacional e processada no mesmo, mantendo o cubo no servidor. Isso permite analisar grandes volumes de dados; entretanto, se uma grande quantidade de usuários acessam esses bancos simultaneamente, pode haver problemas de desempenho no servidor.

Esta arquitetura tem a vantagem de utilizar tecnologia estabelecida, de arquitetura aberta e padronizada, beneficiando-se da diversidade de plataformas, da escalabilidade e do paralelismo de *hardware*; também possui a vantagem de não restringir o volume de armazenamento de dados. Sua desvantagem é o conjunto pobre de funções para análises dimensionais e o baixo desempenho da linguagem SQL na execução de consultas pesadas. (CARVALHO, 2004).

2.7.2.3 HOLAP (HYBRID ON LINE ANALYTICAL PROCESSING)

A arquitetura HOLAP (*Hybrid On Line Analytical Processing*) possui uma forma híbrida de acessar os dados, ou seja, uma combinação entre ROLAP e MOLAP. Dessa forma, é possível combinar o melhor dos dois mundos, isto é, combinar o alto desempenho do MOLAP com a alta escalabilidade do ROLAP.

Esta arquitetura tem a capacidade de gerenciar grandes quantidades de dados a partir de implementações ROLAP, somadas ao alto desempenho típico das consultas MOLAP. Mais especificamente, HOLAP condiz com a ideia de que a maior quantidade de dados deve ser armazenada em um SGBDR para evitar os problemas causados pela dispersão, enquanto que o SGBDM armazena somente as informações que os usuários finais frequentemente precisam acessar. No caso de as informações serem insuficientes,

o sistema, de forma transparente ao usuário final, acessa a parte dos dados gerenciados pelo SGBDR (CARVALHO, 2004).

2.7.2.4 DOLAP (DESKTOP ON LINE ANALYTICAL PROCESSING)

A arquitetura DOLAP que é uma arquitetura *desktop* do OLAP, ou seja, é uma ferramenta para usuários que possuam uma cópia da base multidimensional ou de um subconjunto dela ou ainda, que queiram acessar um repositório de dados central localmente (MACHADO, 2008). O usuário, ao acessar este repositório, dispara uma instrução SQL e acessa os cubos já existentes no banco de dados multidimensional residente no servidor OLAP e obtém de volta uma informação para ser analisada em sua estação de trabalho.

As vantagens desta arquitetura são: a diminuição do tráfego na rede, a não sobrecarga do servidor de banco de dados e a diminuição de problemas de escalabilidade. Porém as desvantagens podem ser que o cubo de dados seja muito grande a ponto de tornar a análise dos dados muito lenta ou mesmo inviável, por conta da configuração da estação cliente.

2.7.2.5 WOLAP (WEB ON LINE ANALYTICAL PROCESSING)

Por fim, a arquitetura *WOLAP* é acessada através de um navegador *web*. Um cliente faz uso do navegador para se comunicar com um servidor *HTTP* (protocolo de comunicação), que por sua vez se utiliza de um *middleware* (programa de computador que faz a mediação entre outros softwares) responsável pela comunicação com o servidor de banco de dados.

A arquitetura WOLAP poderá ser a chave para aplicações na internet e disponibiliza um caminho simples e barato no acesso à dados do *Data Warehouse*. A utilização de *Web browsers* para acesso OLAP está sendo divulgados, mas poucos sites a utilizam. Esta ferramenta é a migração de OLAP para internet.

Esta arquitetura é utilizada de uma ferramenta OLAP a partir de um *browser*. Possui duas tecnologias em constante evolução: a primeira é a *Web* e a segunda são as ferramentas OLAP. A diferença entre esta ferramenta e as outras é que ela utiliza a *Web*, facilitando, assim, a distribuição da ferramenta, o acesso remoto dos dados a serem analisados e a utilização da aplicação independente de plataforma (ANDREATO, 1999).

2.8 CONSIDERAÇÕES FINAIS

Neste Capítulo foram apresentados os conceitos de *Business Intelligence*, que surgiu para facilitar no processo de decisões empresariais, a partir de informações e análises dos dados. O capítulo seguinte trata da descrição geral do estudo de caso na empresa Samara calçados, contendo seus passos no processo através da utilização do *QlikView* uma das ferramentas OLAP.

3 ESTUDO DE CASO

Neste capítulo serão aplicados os conceitos de implantação de uma solução BI na Empresa Samara Calçados, utilizando a ferramenta *QlikView* 10 para demonstrar a criação de uma solução BI.

3.1 A EMPRESA

A Samara Calçados atua desde 1977 na produção de sapatinhos de bebês. Encontra-se sob a responsabilidade dos irmãos José Saad Rached e Antonio Saad Rached. Com o passar do tempo, os sócios investiram cada vez mais na expansão e no desenvolvimento da empresa, com foco em uma estrutura física mais próxima de uma fábrica, transformando cada ação em uma parcela de contribuição para que esse sonho se realizasse.

Hoje, com mais de 100 funcionários trabalhando em diversos turnos e funções, a Samara Calçados produz uma média de 30.000 pares de calçados por mês, fornecendo produtos para todo o Brasil, a Argentina, o Uruguai, a Itália entre, outros países.

A Samara tem como missão fabricar calçados infantis oferecendo ao mercado produtos com um nível de qualidade que venha a atender ou superar as expectativas de seus clientes. Também tem como missão proporcionar um ambiente seguro e com condições adequadas, de forma a satisfazer seus profissionais e colaboradores.

Com o crescimento da empresa, surgiu a necessidade de se obter, em um tempo hábil, informações mais elaboradas sobre os setores da empresa, solução inteligente para conseguir conquistar um público ainda maior. A implantação de *Business Intelligence* vem para inovar a tomada de decisões na empresa, diante de resultados.

Atualmente, as empresas de pequeno, médio e grande porte buscam sistemas que proporcionem rapidez, flexibilidade e facilidade de uso. Foi utilizada, nesta empresa, uma ferramenta OLAP chamada *QlikView*, que possui a capacidade de transformar dados em conhecimento do negócio, permitindo melhores decisões e criando novas oportunidades para qualquer tipo de empresa.

3.2 QLIKVIEW

O *QlikView* que é um *software* de *Business Intelligence*, que foi desenvolvido pela empresa sueca *QlikTech*, fundada no ano de 1993, como uma empresa de consultoria. Um dos primeiros trabalhos da empresa era desenvolver uma ferramenta para análise de dados multidimensional. Ao desenvolver esta solução, os fundadores perceberam que poderiam utilizar esta ferramenta para solução de problemas gerais de outras organizações. Isto levou ao desenvolvimento da ferramenta *Qlikview* (VALE, 2010).

O *QlikView* é um *software* inovador, que utiliza uma tecnologia e uma linguagem própria para análise de informações relacionais provenientes de qualquer fonte de dados. Um *software* centrado na simplificação de tomada de decisão para usuários de negócios entre as organizações. O *QlikView* é a primeira plataforma de *Business Intelligence* associativa *in-memory*. É capaz de fazer associações ao ligar dados de diversas fontes com apenas alguns cliques, parecido como a nossa mente.

No *QlikView* todos os dados relevantes em todas as dimensões ficam disponíveis em memória RAM. Estas associações de dados são feitas pela tecnologia AQL (acrônimo de *Associative Query Logic*), patenteada pela *Qliktech*, uma linguagem muito parecida com a linguagem SQL (BUSTAMANTE, 2006).

A tecnologia AQL promove análises não hierárquicas de dados, garante assim análises onde as consultas não foram construídas em uma ordem específica, por exemplo, não é necessário totalizar valores mensais ou anuais. Usando AQL, qualquer valor da estrutura de dados pode ser o ponto de partida para a análise.

As aplicações do *QlikView* podem ser geradas livremente e, muitas vezes, sem a dependência da área de TI, a partir da necessidade de cada usuário, tendo como foco as

informações e dados que são determinados pelo usuário final, que podem ser análises de desempenho de vendas, análises financeiras, planejamento de recursos corporativos ou qualquer outro tipo de visão. O usuário tem total liberdade para defini-lá.

A edição mais avançada e com capacidade expandida para comportar bilhões de registros é a versão 10 em português, que atende de forma eficaz, às necessidades dos usuários em plataforma 64 bits ou 32bits.

No BI tradicional, a solução é construída em camadas de cubos OLAP intermediária entre os dados e a interface com o usuário; já, no *QlikView*, ela é extraída de forma automática pela sua própria interface. A principal diferença entre estas soluções é a forma como o *QlikView* se relaciona com o banco de dados.

Escolheu-se o *Qlikview* por conta da sua facilidade de elaboração de relatórios pelos usuários finais, pois, após ser realizado o processo de ETL, os relatórios podem ser produzidos com inclusão de elementos gráficos e conhecimentos básicos de funções de agregação, somas e contagens por exemplo.

3.3 APLICAÇÃO DO BI NA SAMARA CALÇADOS

A Empresa Samara Calçados utiliza os dados em um banco de dados operacional e os relatórios são gerados pelos setores de produção, financeiro, vendas entre outros.

Uma vantagem do sistema BI para empresa é a facilidade da criação de relatórios e pesquisas específicas relativas às necessidades emergentes dos gestores.

Para realizar a implantação do modelo BI foi utilizada a ferramenta *QlikView*, com que foi realizado todo o processo de ETL, através de um *backup* do banco de dados do dia 06 de outubro de 2011.

3.3.1 Base de Dados da Empresa

A fim de representar os benefícios que o *Business Intelligence* pode oferecer a empresa, foi estudada a base de dados da empresa onde foram observadas as necessidades para compor o protótipo do estudo de caso. Motivado pela necessidade de manter a atualização diária da base de dados que armazena dados de forma a tornar precisa e confiável a prestação de serviços, no que diz respeito ao controle gerencial para tomada de decisões.

A Samara Calçados depende totalmente do bom funcionamento do sistema OLTP para realizar o seu trabalho de forma eficiente, onde este não pode parar e nem ter falhas de grande importância. O SGBD utilizado é o *SQL Server*, possuem 52 (cinquenta e duas) tabelas, com 389.527 mil registros, onde esta apresenta falta de informações necessárias para tomada de decisões (por causa de tabelas com muitos registros), dados inconsistentes, tabelas com muitas colunas, com muitos registros e dados redundantes.

A base de dados é alimentada todos os dias no final do expediente através do *backup* do sistema, pois o BI trabalha com um dia anterior, pois as informações necessárias ficam na memória.

As tabelas selecionadas abaixo foram selecionadas para a construção do repositórios de dados conhecido como DW, essas tabelas são de suma importância pois contém dados necessários para a análise das informações, tendo em vista que o intuito da empresa em saber de informações detalhadas sobre um determinado produto, ou sobre um cliente, deve ter informações de tabelas completas com estes dados.

A tabela 1 descreve a quantidade de campos e número de registros constatados em cada tabela do banco de dados, como também, os respectivos totais de registros. Estas são as tabelas selecionadas para o processo de *Business Intelligence*.

TABELA 1: Informações sobre as principais tabelas do sistema.

Principais Tabelas	Quantidade de Campos	Número de Registros
Clientes e Fornecedores	12	4.758
Produtos	02	181.992
Pedidos	04	192.372
Setor	03	23
Insumo	02	9.678
Pagamento	03	704
TOTAL	26	389.527

FONTE: Autor próprio.

No que se refere à descrição de cada coluna das tabelas descritas em tabela e ao valor dos dados, as informações da seguinte forma: nome da tabela, nome das colunas, descrição que define o nome campo e, por último, o valor dos dados que se mostraram presentes em cada campo.

A seguir serão apresentadas as descrições das principais tabelas.

TABELA 2: Tabela Clientes e Fornecedores.

Tabelas	Campo	Descrição do campo
Clientes e Fornecedores	Clidcod	Código do cliente ou fornecedor
	Clidtip	Tipo do cliente ou fornecedor
	Clinom	Nome do cliente ou fornecedor
	Cliend	Endereço do cliente ou fornecedor
	Clibai	Bairro do cliente ou fornecedor
	Clicid	Cidade do cliente ou fornecedor
	Clidest	Estado do cliente ou fornecedor
	Clicep	Cep do cliente ou fornecedor
	Clifon	Telefone do cliente ou fornecedor
	Clifax	Telefone fax do cliente ou fornecedor
	Cliegcc	Cnpj ou Cgc do cliente ou fornecedor
	Cliinsc	Inscrição do cliente ou fornecedor
	Climail	Email do cliente ou fornecedor
Clipais	País de origem do cliente ou fornecedor	

FONTE: Autor próprio.

Nesta tabela foram dispostas as informações da tabela de clientes e fornecedores, que no banco de dados eles só se diferem pelo tipo.

TABELA 3: Tabela Produtos.

Tabelas	Campo	Descrição do campo
Produtos	Prodid	Código de identificação do produto
	Proddesc	Descrição do produto

FONTE: Autor próprio.

Esta tabela de produtos esta identificando e descrevendo os produtos.

TABELA 4: Tabela Pedidos.

Tabelas	Campo	Descrição do campo
Pedidos	Pedcod	Código do pedido
	Peditem	Item do pedido
	Pedmoddesc	Descrição do modelo
	Pedtotind	Valor total do pedido

FONTE: Autor próprio.

Nesta tabela estão dispostas informações sobre o pedido como: código do pedido, itens do pedido, descrição de modelos dos produtos e o valor total do pedido.

TABELA 5: Tabela Setor.

Tabelas	Campo	Descrição do campo
Setor	Setcod	Código do setor
	Setnom	Nome do setor
	Setabr	Nome abreviado do setor

FONTE: Autor próprio.

A tabela setor contém o código, nome, e abreviação dos setores de produção da empresa.

TABELA 6: Tabela Insumo.

Tabelas	Campo	Descrição do campo
Insumo	Procod	Código da descrição dos produtos de matéria prima
	Reqprodesc	Descrição dos produtos de matéria prima

FONTE: Autor próprio.

Nesta tabela de insumo apresenta o código descrição dos produtos de matéria prima, e sua descrição.

TABELA 7: Tabela Pagamento.

Tabelas	Campo	Descrição do campo
Pagamento	Pagcod	Código da forma de pagamento
	Pagdesc	Descrição da forma de pagamento
	Pagdia	Descrição de datas

FONTE: Autor próprio.

Nesta tabela de pagamento apresenta o código, a descrição e as datas da forma de pagamento.

3.3.2 Processo de ETL no QlikView

O processo de extração começa pela conexão ao banco de dados, logo após inicia a criação de *scripts* de extração de dados, realizando a busca pela informação da base de dados da empresa. O *script* é composto basicamente de comandos *select* puros (sem modificar a estrutura das tabelas) que após ser executado gera um arquivo QVD utilizando o comando *store* do *QlikView*, este arquivo é uma cópia dos dados da tabela do banco de dados criptografada, em seguida é utilizado o comando *drop table* para remoção dos dados da memória.

QUADRO 1: Exemplo de extração da tabela.

```
12 ODBC CONNECT TO SAMMARA;
13
14 consped: /*Nome da Tabela*/
15
16 /*Realiza a carga dos dados*/
17 SQL SELECT * FROM incca.dbo.consped;|
18
19 /*Gera a tabela na extensão da linguagem .qvd*/
20 STORE consped INTO consped.QVD;
21
22 /*Remove a tabela da memória*/
23 DROP TABLE consped;
24
```

FONTE: Autor próprio.

Após o processo de extração pronto, começa a etapa de transformação dos dados, criação do modelo dimensional e as associações entre as tabelas. Nesta fase da etapa é como o processo de extração utiliza os comandos AQL que são bastante parecidos com os comandos do SQL, como demonstrado a seguir.

QUADRO 2: Exemplo de transformação dos dados.

```
1 DimCliente:
2
3 LOAD clicod as DimCliente.IdCliente, /* Renomeando as Chaves*/
4     clinom as DimCliente.Nome,
5     cliend as DimCliente.Endereco,
6     clibai as DimCliente.Bairro,
7     clicid as DimCliente.Cidade,
8     cliest as DimCliente.Estado,
9     cliccep as DimCliente.CEP,
10    clifon as DimCliente.Telefone,
11    clifax as DimCliente.Fax,
12    clicgc as DimCliente.CGC,
13    cliconta as DimCliente.Conta,
14    clifan as DimCliente.NomeFantasia,
15    climail as DimCliente.Email,
16    clipais as DimCliente.Pais
17 FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\extração\p0012.QVD] (qvd);
18
19 STORE DimCliente INTO Dimensoes/DimCliente.qvd; /* Criando a DimCliente*/
20
21 DROP Table DimCliente; /* Remover a tabela da memória*/
22
```

FONTE: Autor próprio.

Na etapa de transformação, cada tabela de dimensão é construída a partir de dados extraídos da base de origem e possui uma chave de identificação e seus respectivos atributos, que serão utilizados como filtro nas consultas realizadas.

As dimensões criadas foram:

- ✓ Dimensão Tempo
- ✓ Dimensão Cliente
- ✓ Dimensão Produto
- ✓ Dimensão Fornecedora
- ✓ Dimensão Setor
- ✓ Dimensão Pagamento
- ✓ Dimensão Insumo

QUADRO 3: Exemplo da dimensão tempo.

```
41 LOAD
42     date(Data, 'YYYYMMDD') AS IdTempo,
43     date(Data, 'DD/MM/YYYY') AS Data,
44     day(Data) AS Dia,
45     WeekDay(Data) AS DiaSemana,
46     Month(Data) AS Mes,
47     date(monthstart(Data), 'YYYY/MM') AS MesAno,
48     num(date(monthstart(Data), 'MM')) AS NuMes,
49     Year(Data) AS Ano,
50     if (Num(Upper(date(monthstart(Data), 'MM'))) <= 3, '1',
51     if (Num(Upper(date(monthstart(Data), 'MM'))) >3 AND Num(Upper(date(monthstart(Data), 'MM'))) <=6, '2',
52     if (Num(Upper(date(monthstart(Data), 'MM'))) >6 AND Num(Upper(date(monthstart(Data), 'MM'))) <=9, '3',
53     if (Num(Upper(date(monthstart(Data), 'MM'))) >9 AND Num(Upper(date(monthstart(Data), 'MM'))) <=12, '4'))))
54     AS Trimestre,
55     if (Num(Upper(date(monthstart(Data), 'MM'))) <= 6, '1', '2') AS Semestre
56 FROM Dimensoes\DimTempo.qvd (qvd);
57
58 UNQUALIFY *;
59
60 STORE DimTempo INTO Dimensoes/DimTempo.qvd;
61
62 DROP TABLE DimTempo;
63
```

FONTE: Autor próprio.

A dimensão tempo é uma das mais importantes, responsável pela manipulação de forma dinâmica por período.

QUADRO 4: Exemplo da dimensão cliente.

```
1 DimCliente:
2 LOAD clicod as DimCliente.IdCliente,
3     clinom as DimCliente.Nome,
4     cliend as DimCliente.Endereco,
5     clibal as DimCliente.Bairro,
6     clicid as DimCliente.Cidade,
7     cliest as DimCliente.Estado,
8     cliccep as DimCliente.CEP,
9     clifon as DimCliente.Telefone,
10    clifax as DimCliente.Fax,
11    clicgc as DimCliente.CGC,
12    cliconta as DimCliente.Conta,
13    clifan as DimCliente.NomeFantasia,
14    climail as DimCliente.Email,
15    clipais as DimCliente.Pais
16 FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\extração\p0012.QVD] (qvd);
17
18 STORE DimCliente INTO Dimensoes/DimCliente.qvd;
19
20 DROP Table DimCliente;
21
```

FONTE: Autor próprio.

A dimensão cliente é responsável por conter informações pessoais como nome, endereço, bairro, cidade, estado, cep, telefone, enfim informações completas.

QUADRO 5: Exemplo da dimensão produto.

```
1 DimProduto:
2 LOAD modcod as DimProduto.IdProduto,
3     pedmoddesc as DimProduto.Descricao
4 FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\extração\p0031.QVD] (qvd);
5
6 STORE DimProduto INTO Dimensoes/DimProduto.qvd;
7
8 DROP Table DimProduto;
9
```

FONTE: Autor próprio.

A dimensão produto é responsável pela identificação do produto e pela descrição.

Quadro 6: Exemplo da dimensão setor.

```
1 DimSetor:
2 LOAD Setor as DimSetor.IdSetor,
3     Setnom as DimSetor.Descricao
4 FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\extração\P0008.QVD] (qvd);
5
6 STORE DimSetor INTO Dimensoes/DimSetor.qvd;
7
8 DROP Table DimSetor;
9
```

FONTE: Autor próprio.

A dimensão setor é responsável pela identificação do setor e descrição.

QUADRO 7: Exemplo da dimensão fornecedor.

```
1 DimFornecedor:
2 LOAD clicod as DimFornecedor.IdFornecedor,
3     clinom as DimFornecedor.Nome,
4     cliend as DimFornecedor.Endereco,
5     clibai as DimFornecedor.Bairro,
6     clicid as DimFornecedor.Cidade,
7     cliest as DimFornecedor.Estado,
8     clicep as DimFornecedor.CEP,
9     clifon as DimFornecedor.Telefone,
10    clifax as DimFornecedor.Fax,
11    clicgc as DimFornecedor.CGC,
12    cliconta as DimFornecedor.Conta,
13    clifan as DimFornecedor.NomeFantasia,
14    climail as DimFornecedor.Email,
15    clipais as DimFornecedor.Pais
16 FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\extração\p0012.QVD] (qvd);
17
18 STORE DimFornecedor INTO Dimensoes/DimFornecedor.qvd;
19
20 DROP Table DimFornecedor;
21
```

FONTE: Autor próprio.

A dimensão fornecedor informa o id do fornecedor, nome, endereço, bairro, cidade, cep, telefone, toas as informações detalhadas.

QUADRO 8: Exemplo da dimensão pagamento.

```
1 DimPagamento:
2 LOAD pagcod as DimPagamento.IdPagamento,
3     pagdes as DimPagamento.Descricao
4 FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\extração\fa4.QVD] (qvd);
5
6 STORE DimPagamento INTO Dimensoes/DimPagamento.qvd;
7
8 DROP Table DimPagamento;
9
```

FONTE: Autor próprio.

A dimensão pagamento identifica e descreve a informação.

QUADRO 9: Exemplo da dimensão insumo.

```
1 DimInsumo:
2 LOAD Nrooc&Itemoc as DimInsumo.IdInsumo,
3     reqprodesc as DimInsumo.Descricao
4 FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\extração\E0002.QVD] (qvd);
5
6 STORE DimInsumo INTO Dimensoes/DimInsumo.qvd;
7
8 DROP Table DimInsumo;
```

FONTE: Autor próprio.

A dimensão insumo demonstra a identificação e descrição dos produtos de matéria prima.

QUADRO 10: Exemplo da tabela fato.

```
1 Fato: /*Nome da tabela*/
2
3 /*Carga dos dados*/
4 LOAD Nrooc as Fato.IdFato,
5     Nrooc&Itemoc as DimInsumo.IdInsumo,
6     date(reqdatdes, 'YYYYMMDD') as DimTempo.IdTempo,
7     Setor as DimSetor.IdSetor,
8     ocvalunit as Fato.ValorUnitarioCompra,
9     reqqt as Fato.QuantidadeCompra,
10    ocvalunit*reqqt as Fato.ValorTotalCompra
11 FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\extração\E0002.QVD] (qvd);
12
13 Left Join (Fato) /*Realiza as junções de tabelas*/
14 LOAD Nrooc as Fato.IdFato,
15     PagCod as DimPagamento.IdPagamento,
16     OcTipoFret as Fato.TipoFrete,
17     Clicod as DimFornecedor.IdFornecedor
18 FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\extração\E0003.QVD] (qvd);
19
20 Concatenate |
21 Load pedcod as Fato.IdFato,
22     date(peddatfat, 'YYYYMMDD') as DimTempo.IdTempo,
23     pedcli as DimCliente.IdCliente,
24     modcod as DimProduto.IdProduto,
25     pedcomissa as Fato.ValorComissao,
26     peddesc1+peddesc2+peddesc3+peddesc4 as Fato.Desconto,
27     pedvalfat as Fato.Valor
28 FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\extração\p0031.QVD] (qvd);
29
30 STORE Fato INTO Fato/Fato.qvd;
31
32 DROP Table Fato;
33
```

FONTE: Autor próprio.

A tabela fato é composta por todas as medidas e chaves estrangeiras das dimensões que fazem parte do modelo multidimensional.

O próximo passo é o carregamento de dados para um único repositório, onde estes já estarão transformados e prontos para a elaboração do painel executivo. Todas as informações serão apresentadas em gráficos e tabelas de maneira simples e interativa.

QUADRO 11: Exemplo de carregamento de dados.

```

8 SET TimestampFormat='DD/MM/YYYY hh:mm:ss[.fff]';
9 SET MonthNames='jan;fev;mar;abr;mai;jun;jul;ago;set;out;nov;dez';
10 SET DayNames='seg;ter;qua;qui;sex;sáb;dom';
11
12 LOAD * FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\transformação\Dimensoes\DimCliente.qvd] (qvd);
13
14 LOAD * FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\transformação\Dimensoes\DimFornecedor.qvd] (qvd);
15
16 LOAD * FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\transformação\Dimensoes\DimInsumo.qvd] (qvd);
17
18 LOAD * FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\transformação\Dimensoes\DimPagamento.qvd] (qvd);
19
20 LOAD * FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\transformação\Dimensoes\DimProduto.qvd] (qvd);
21
22 LOAD * FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\transformação\Dimensoes\DimSetor.qvd] (qvd);
23
24 LOAD * FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\transformação\Dimensoes\DimTempo.qvd] (qvd);
25
26 LOAD * FROM [C:\Users\windows\Desktop\samara atualizada\app_qv\transformação\Fato\Fato.qvd] (qvd);
27

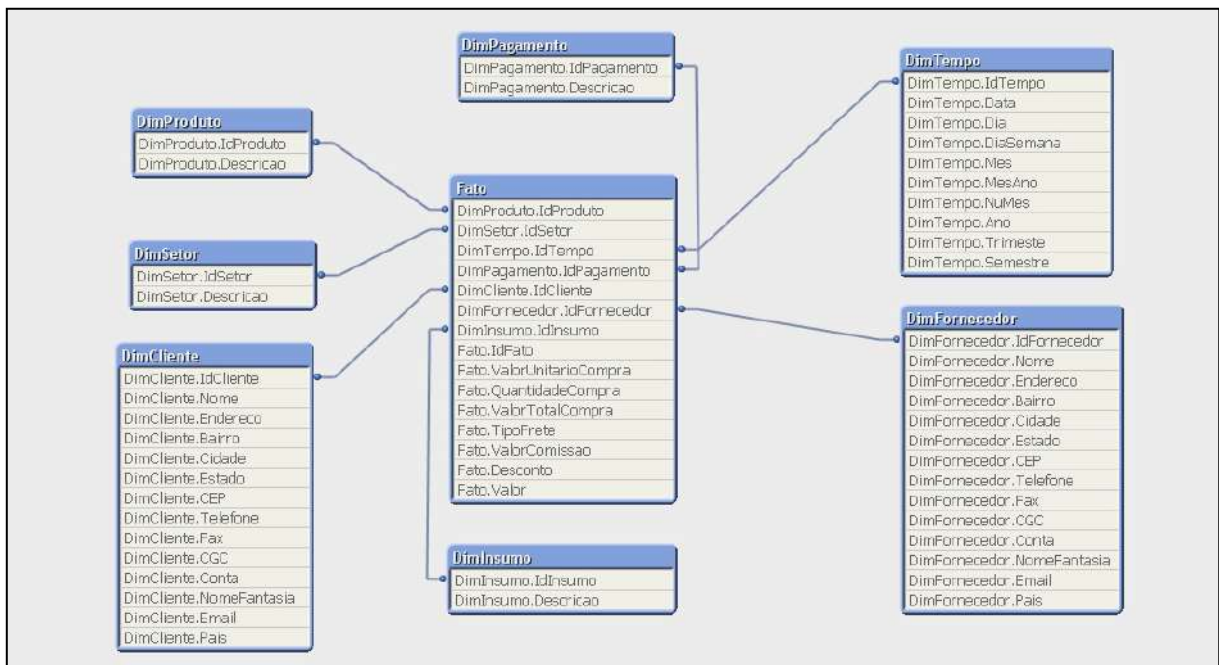
```

FONTE: Autor próprio.

É realizado o carregamento das informações da transformação, para visualização dos painéis.

Na figura a seguir será apresentado o modelo multidimensional estrela do projeto, conforme foi escrito na secção.

FIGURA 10: Modelo Multidimensional.



FONTE: Autor próprio.

O modelo multidimensional demonstrado é composto por uma tabela fato com suas sete dimensões sendo estas: dimensão produto, dimensão setor, dimensão cliente, dimensão pagamento, dimensão insumo, dimensão tempo e dimensão fornecedor.

A visualização das tabelas é importante para analisar e validar o modelo, para evitar possíveis falhas, verificando se não houve falta de informação de uma determinada tabela.

3.3.3 Construção de Cenários

Para atender as necessidades da empresa Samara Calçados, foram construídos seis painéis, sendo uma visão geral, o faturamento, compras, tendências, produtos, pagamentos, utilizando componentes como gráficos e seletores para apresentar as informações ao usuário final.

O painel executivo para ser montado foi inserido a logomarca da empresa, colocado seletores de ano e mês que a partir dele o usuário pode selecionar o mês de visualização e o ano, podendo também fazer o comparativo entre meses e anos.

FIGURA 11: Painel Executivo Visão Geral.



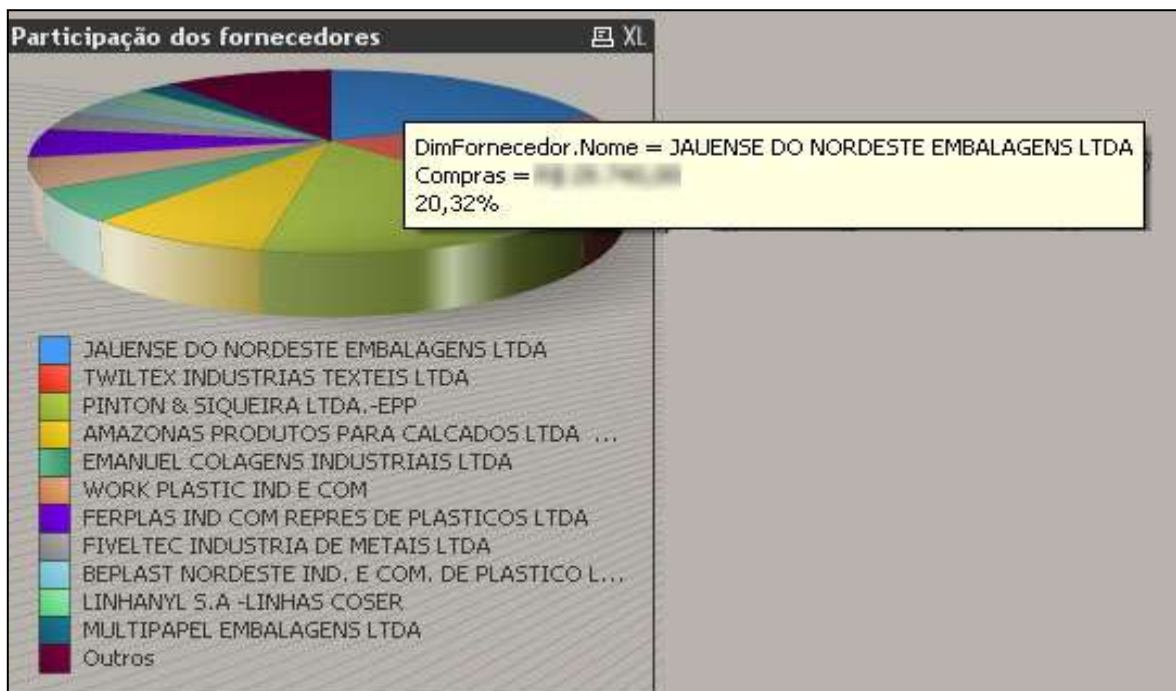
FONTE: Autor próprio.

Nesta aba do painel executivo, foi criado um gráfico evolução, no qual descreve mês a mês as vendas e as compras, onde demonstra valores. O segundo gráfico mais abaixo do lado esquerdo é o de cliente mais frequentes que mostra os cinco melhores clientes de acordo com o ano e o mês selecionado.

O gráfico do meio, participação dos fornecedores demonstra a sua parcela de fornecimento para a empresa, na ferramenta colocando o mouse acima esta parte da

fatia do gráfico de pizza mostra o valor de compras de acordo com seletores de mês e ano.

FIGURA 12: Exemplo do valor Demonstrado ao ser selecionado.



FONTE: Autor próprio.

O exemplo acima do gráfico de pizza foi selecionado o ano 2010 e os meses de janeiro a junho, o valor total demonstrado corresponde á fatia do gráfico.

A próxima tela a seguir é a visão do faturamento da empresa.

FIGURA 13: Painel Executivo Faturamento.



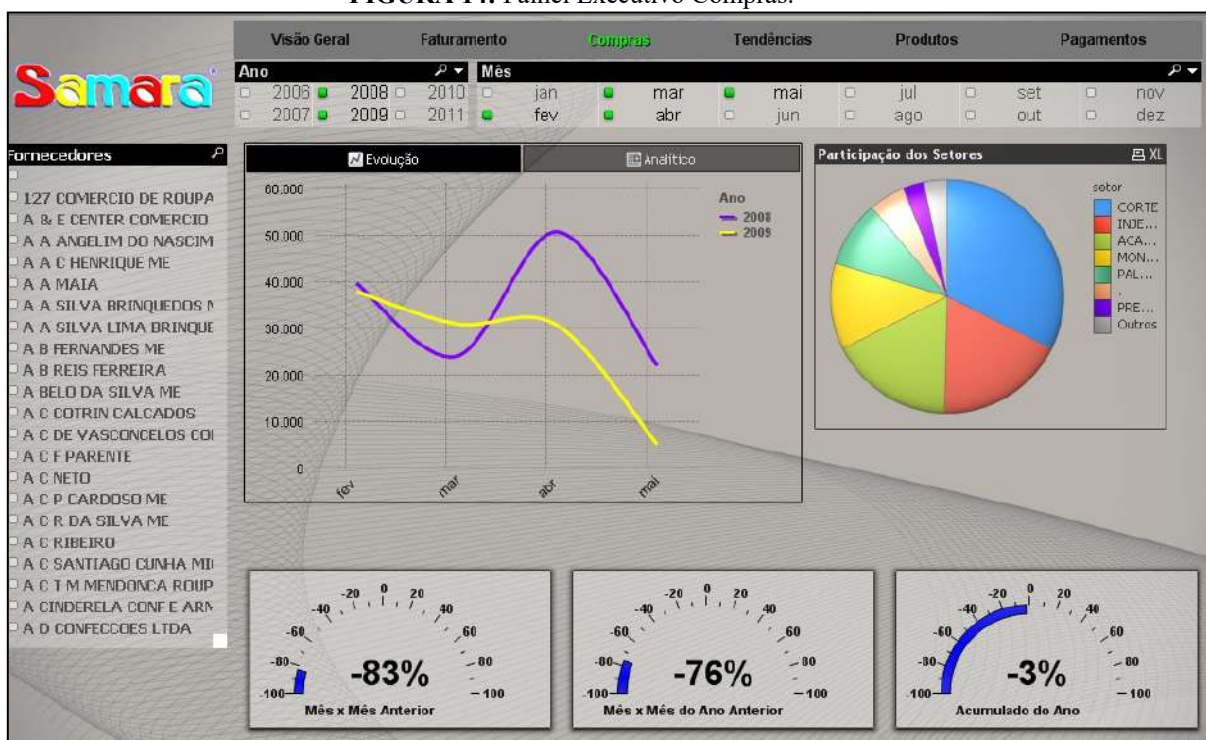
FONTE: Autor próprio.

Na aba faturamento os anos de 2010 e 2011 selecionados e os meses de março, abril e maio para demonstrar que os velocímetros compara o mês atual com o mês anterior, o segundo velocímetro compara o ano atual com o ano anterior e no terceiro velocímetro demonstra o percentual acumulado do ano.

Os seletores desta aba são clientes, ano e mês. O gráfico de sazonalidade semanal demonstra os percentuais de faturamento dos dias da semana. O gráfico ao lado sazonalidade mensal onde estão selecionados três meses e mostra em percentuais mensais.

O ultimo gráfico da evolução tem uma linha curva com indicadores de valores sobre o faturamento da empresa. A próxima aba será das Compras.

FIGURA 14: Painel Executivo Compras.



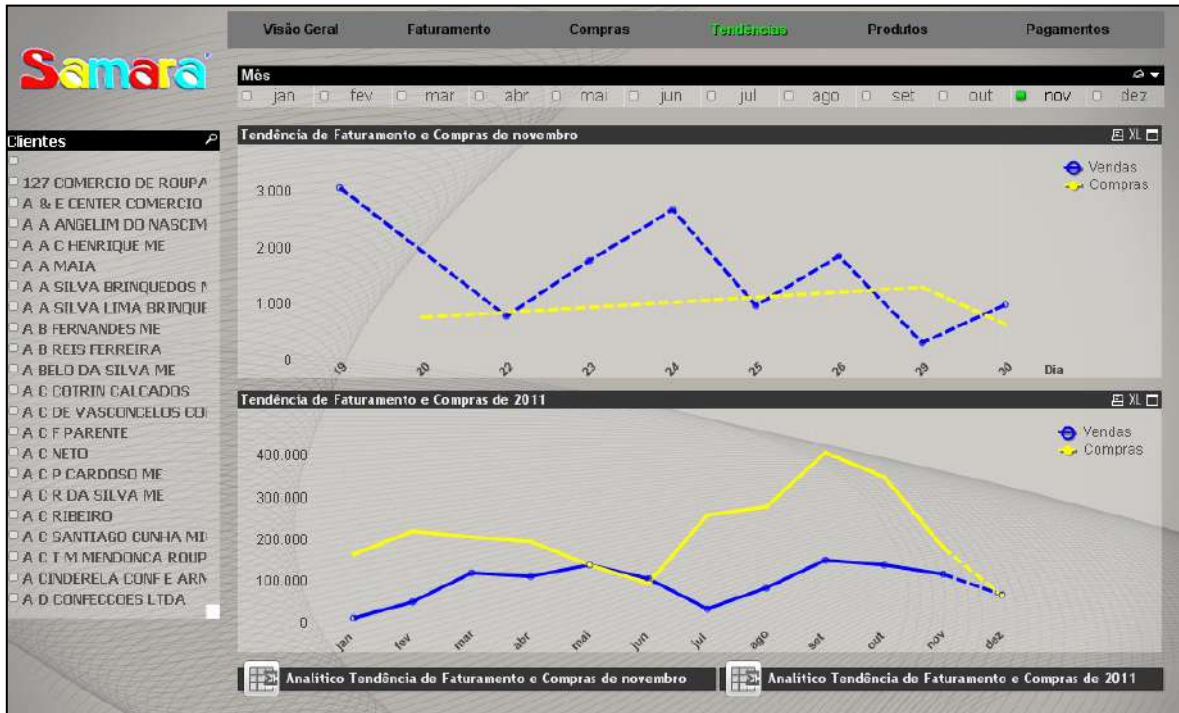
FONTE: Autor próprio.

A tela do painel executivo denominado compras, contém seletores fornecedores, ano, meses, estão selecionados para demonstração os anos 2008 e 2009, e os meses de fevereiro até maio.

O gráfico da evolução do lado esquerdo mostra os valores alcançados e a participação dos valores dos setores da empresa, mais abaixo se encontram percentuais relativos ao mês atual com mês anterior, mês com mês do ano anterior e o acumulado do ano.

A próxima aba do painel executivo é bem interessante para o *Business Intelligence*, pois nela contém informações futuras.

FIGURA 15: Painel Executivo Tendências.



FONTE: Autor próprio.

Nesta aba o gráfico demonstra como poderá ser as vendas e as compras baseadas em informações anteriores, ou seja, prever algo para os gestores de acordo com a evolução da empresa.

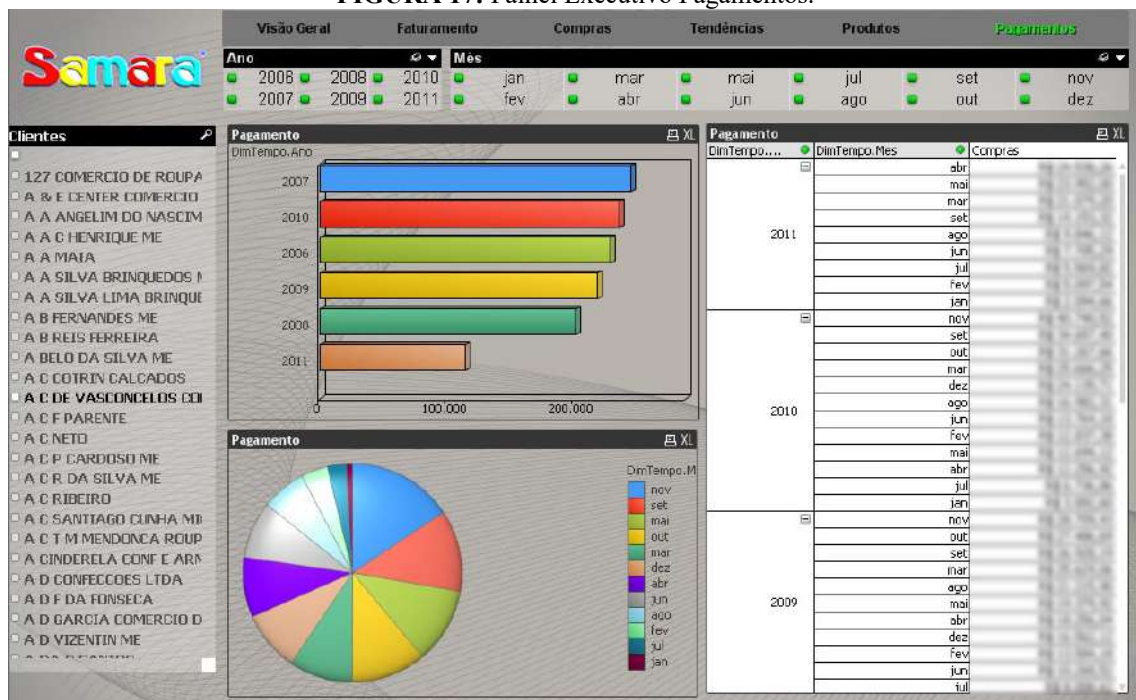
FIGURA 16: Painel Executivo Produtos.



FONTE: Autor próprio.

A aba Produtos demonstra o produto mais vendido no ano de 2011, nos meses de janeiro até agosto, contendo uma barra de rolagem para verificar o menor.

FIGURA 17: Painel Executivo Pagamentos.



FONTE: Autor próprio.

Nesta aba o gráfico demonstra como poderá ser as vendas e as compras baseadas em informações anteriores, ou seja, prever algo para os gestores de acordo com a evolução da empresa.

3.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Neste capítulo o intuito foi demonstrar o funcionamento do *Business Intelligence* na empresa Samara Calçados para estudo de caso, a funcionalidade da ferramenta *QlikView*, e todos os passos do processo.

4 RESULTADO E CONCLUSÃO

Este trabalho apresentou um estudo sobre a aplicação de *Business Intelligence* na empresa Samara calçados, muito utilizados atualmente para a tomada de decisões nas empresas, com o intuito de auxiliar aos gestores ou até mesmo as organizações há entenderem um pouco melhor sobre estes paradigmas. Neste sentido, busca auxiliá-los ao gerenciamento das empresas, com a intensão de se destacarem no mercado.

Primeiramente este trabalho concentrou-se em estudar fundamentos teóricos sobre o tema abordado, principalmente no que se diz respeito aos processos de tomada de decisões e suas etapas.

Posteriormente, foi desenvolvido o processo de BI com o auxílio de ferramentas OLAP como a *QlickView*, o que tornou de forma simplista a manipulação feita pelos

gestores, pois o software utilizado tem semelhança com a *web*, de fácil seleção e se mostrou eficaz e preventivo, demonstrando tendências futuras.

A solução apresentada à empresa Samara Calçados foi de forma satisfatória, pois informações como: evolução anual, tendências futuras, os relatórios da empresa não eram capazes de demonstrar. Encontrou-se certa dificuldade em relação ao banco de dados, na identificação das tabelas, pois o banco não tinha um dicionário de dados e suas tabelas não tinham nome eram apenas códigos.

Após a realização deste trabalho, conclui-se que é perfeitamente viável o processo de *Business Intelligence* para uma empresa, constatando que seu uso irá acrescentar inúmeros benefícios à empresa, principalmente no que diz respeito à tomada de decisões.

Para finalizar, fica a sugestão para trabalhos futuros a aplicação do processo *Business Intelligence* envolvendo outras ferramentas.

5 REFERÊNCIAS

ANDREATTO, R. **Construindo um Data Warehouse e Analisando suas Informações com Data Mining e OLAP**. Monografia Final de Curso. Faculdade de Ciências Administrativas, Faculdade de Valinhos. 1999.

BARBIERI, Carlos. **Business Intelligence: Modelagem e Tecnologia**. Rio de Janeiro: Axcel Books. 2001.

BUSTAMANTE, L. **QlikView - Uma análise parcial**. Disponível em:<<http://imasters.com.br/artigo/4993/gerencia-de-ti/qlikview-uma-analise-parcial>>. Acesso em: 20 nov. 2011.

CARVALHO, B.F. **Arquiteturas de Ferramentas OLAP**. SQL Magazine, Rio de Janeiro, ano 1, ed. 9, p.12-16, 2004.

COSTA, A. M. V. X. **BUSINESS INTELLIGENCE – BI: Uma Aplicação de Business Intelligence na Advocacia**. João Pessoa. Unipê. 2011.

FELBER, Edmilson J. W. **Proposta de uma ferramenta OLAP em um Data Mart Comercial: Uma aplicação prática na indústria calçadista**. Trabalho de Conclusão de Curso. Centro Universitário FEEVALE. Novo Hamburgo. 2005.

GOLFARELLI, M.; RIZZI, S. **Data Warehouse Design: modern principles and methodologies**. New York: McGraw-Hill, 2009.

INMON, W. H. **Como construir o Data Warehouse**. Rio de Janeiro: Editora Campus, 1997.

INMON, William. **Building the Data Warehouse**. Indianapolis, Estados Unidos: Editora Wiley Publishing, Inc., 2005.

INTELIGÊNCIA NEGÓCIOS. **A Solução QlikView**. Disponível em:<<http://www.inteligencia-negocios.com/tecnologia>>. Acesso em: 05 nov. 2011.

