**GIULIANO HENRIQUE COSTA**

**HMA**

**SOFTWARE DE GESTÃO PARA BARES E RESTAURANTES**

Trabalho de Conclusão de Curso

apresentado ao Programa de Aplicação

Profissional como requisito parcial para

obtenção do título de Tecnólogo em

Análise e Desenvolvimento de Sistemas

pelo Centro Tecnológico da Universidade

Positivo.

Orientador: Prof. Allston Wagner Silviero Martins

**CURITIBA**

**2015**

**SUMÁRIO**

**INTRODUÇÃO................................................................................................... 4**

**1. FUNDAMENTAÇÃO TEÓRICA..................................................................... 6**

1.1 Gerenciamento de Projetos...................................................................... 6

1.2 Git hub...................................................................................................... 6

1.3 Kanban..................................................................................................... 7

1.4 Estrutura Analítica do Projeto (EAP)........................................................ 8

1.5 Iterativo Incremental................................................................................. 8

1.6 Casos de Uso........................................................................................... 9

1.7 Arquitetura em Três Camadas (3 TIER)................................................... 9

1.8 Microsoft Visual Studio........................................................................... 10

1.9 C# (C Sharp)........................................................................................... 10

1.10 Designer Patterns................................................................................. 10

1.11 Singleton............................................................................................... 11

1.12 Windows Presentation Foundation (WPF) ........................................... 11

1.13 Model View Controller (MVC) .............................................................. 12

1.14 Object-Relational Mapping (ORM) ...................................................... 12

1.15 Frameworks.......................................................................................... 12

1.16 Sql Server Express 2012...................................................................... 13

1.17 Testes................................................................................................... 13

**2. ORGANIZAÇÃO CLIENTE.......................................................................... 14**

**3. DIÁGNOSTICO DO AMBIENTE.................................................................. 15**

**4. OBJETIVOS................................................................................................. 16**

**5. DESENVOLVIMENTO................................................................................. 17**

**6. CONSIDERAÇÕES FINAIS......................................................................... 19**

**REFERÊNCIAS BIBLIOGRÁFICAS................................................................ 20**

**ANEXOS**

**INTRODUÇÃO**

Com o crescimento do consumo em estabelecimentos de entretenimento noturno, inúmeros empreendimentos surgem a cada dia, todos tentando buscar sua fatia de mercado.

Com essa demanda, a busca por uma gestão apurada, um atendimento rápido e eficaz, a qualidade nos serviços prestados, torna-se obrigatório. Ainda mais em um tempo que os clientes não se fidelizam mais a um específico ponto.

Baseando-se nisto, o desenvolvimento de uma aplicação que possa facilitar a gestão independente de onde o responsável esteja, um sistema que possa demonstrar de forma clara resultados gerenciais, torna-se de grande valia.

Demonstrarei no decorrer deste trabalho, a solução necessária para a criação de uma aplicação que possa ser utilizada, mes

Para solucionar o problema, primeiro é preciso analisar e entender como funciona a empresa, compreender a regra de negócio, definir as partes interessadas e apresentar uma solução.

Com esses conceitos de solução e aplicabilidade, pensou em desenvolver uma aplicação, que visa colaborar com o gerenciamento da empresa, cujos processos são realizados manualmente.

O sistema é considerado pronto para o uso quando o usuário efetua o login com conta cadastrada. Os acessos aos módulos são permitidos de acordo com o perfil de cada usuário. O sistema possui por padrão um usuário máster, o mesmo é capaz de criar novos usuários e atribuir os perfis de acordo com a necessidade.

Para escrever e apresentar a aplicação *Hermes Management Assistant,* o presente trabalho é composto por 5 capítulos. O primeiro faz-se um resumo das métodos usadas, ferramentas, linguagens e modelos para o desenvolvimento. O segundo capítulo aborda a organização cliente, conta um pouco a história do cliente. O terceiro faz-se um resumo sobre os processos utilizados na empresa cliente, contando como é o dia a dia. Após uma análise do capítulo anterior, no quarto apresentamos os objetivos e definições. E finalmente o último capítulo, apresentamos o desenvolvimento, é realizado um detalhamento do processo de elaboração e execução.

1. **FUNDAMENTAÇÃO TEÓRICA**

No desenvolvimento de um software, são muitas as tecnologias e recursos existentes. A escolha por uma tecnologia, bem como uma linguagem de programação dentre as disponíveis, exige uma análise criteriosa sobre a finalidade do sistema. Neste capitulo serão abordadas as tecnologias utilizadas ao longo do desenvolvimento deste projeto bem como as justificativas para suas escolhas.

* 1. **Gerenciamento de Projetos**

Segundo a NBR ISO 10006 (2000), projeto é um “processo único, constituído de um grupo de atividades coordenadas e controladas com datas para início e termino para alcançar objetivos conforme requisitos específicos, incluindo limitações de tempo, recurso e custos”. Esta afirmação concorda que o projeto busca um único resultado, com início e fins bem definidos.

Desta forma, torna-se evidente a importância do planejamento e estruturação do projeto, pois evita desperdício de tempo e recursos assim como assegura o atendimento das expectativas do cliente.

* 1. **Github**

O *Git* nasceu, quando o DVCS chamado *BitKeeper* desfez o contrato com o Linux, foi onde surgiu a ideia de fazer uma própria ferramenta, com o que tinham aprendido usando o *BitKeeper*. Criado em 2005, o sistema foi evoluindo e amadurecendo a ponto de se tornar um sistema de fácil de usar. Extremamente rápido e eficiente com grandes projetos, possuí um branching para desenvolvedores não-lineares.

É um versionador de arquivos, utilizado para gerenciar versões de software onde pode-se implementar novas funcionalidades, e sempre vai haver um histórico das operações, sendo possível retroceder, e todos os usuários podem enviar atualizações, quando enviadas as alterações, a versão anterior fica salva, e o administrador escolhe se aquela atualização pode ou não ser efetuada.

A princípio usa a mesma ideia de repositório local, a diferença é que está hospedado em servidores da internet, quase todas as operações usam recursos e arquivos locais, onde o computado já possua um histórico dos projetos, as operações serão instantâneas, pois não é requisitado ao servidor e sim ao banco de dados local, com o *Git* pode trabalhar off-line realizando commits, e quando se reestabelecer a conexão um simples upload, em outros sistemas isso é quase impossível.

*Git* tem sua integridade, que é *checksum* (valor para a integração de integridade), nenhum dado pode ser alterado ser que o *Git* tenha conhecimento, caso tenha algum arquivo corrompido ou perdido o sistema avisara, mas isso é praticamente impossível de acontecer.

* 1. **Kanban**

Criado pelo *Taiichi Ohno*, que se inspirou nos supermercados americanos, onde quando se esvaziava um produto era reposto somente aquela quantidade, pois não havia muito espaço. Desenvolvido para organizar os processos das empresas, tem o objetivo de controlar a movimentação e produção.

O *Kanban* permite um controle mais detalhado da produção com as informações de *quando, quanto e o que produzir*, o método pode ser aplicado para sistemas de produção repetitiva, segundo o criador o *Kanban* tem que respeitar algumas regras:

“*O processo subsequente deve retirar, no processo precedente, os produtos necessários nas quantidades certas e no tempo correto;*

*O processo precedente deve produzir os seus produtos nas quantidades requisitadas pelo processo subsequente;*

*Nenhum item pode ser produzido ou transportado sem um kanban;*

*Produtos com defeito não devem ser enviados ao processo seguinte;*

*O número de kanbans deve ser minimizado continuamente.”*

O *Kanban* é uma ferramenta de metodologia ágil as vezes usado juntamente com o *Scrum*, que tem o objetivo de aumentar a produtividade.

* 1. **Estrutura Analítica do Projeto (EAP)**

Uma EAP (Estrutura Analítica do Projeto) ou em inglês WBS (Work Breakdown Structure) é utilizada para evidenciar as entregas do projeto. Ele inclui todo o escopo do projeto, ou seja, o trabalho necessário para terminar o projeto e atender os requisitos das partes interessadas.

Segundo PMBOK 4

*A EAP é uma decomposição hierárquica orientada*

*á entrega do trabalho a ser executado pela equipe*

*para atingir os objetivos do projeto e criar as entregas*

*requisitadas, com cada nível descendente da EAP*

*representando uma definição gradualmente mais detalhada*

*do trabalho do projeto. É finalizada pela determinação*

*de contas de controle (ver PMBOK). Esses identificadoes*

*produzem uma estrutura para sumarização hierárquica*

*de custos, cronograma e informações sobre recursos.*

O primeiro retângulo da EAP, contém o nome do projeto. Os níveis mais a baixo representam “o quê” será entregue. Os níveis somados, representarão o projeto como um todo.

* 1. **Iterativo Incremental**

Incremental é realizado um planejamento, onde a estratégia é desenvolver várias partes em paralelas e integrar quando estão prontas. Seu objetivo é desenvolver o sistema somente com uma integração.

Iterativo é onde partes do sistema, melhorias e revisão são pré-definidos, que funciona com o incremental, pois no incremental as saídas não são pré-definidas, e o teste do usuário não são utilizados para revisão, já no iterativo sua saída é utilizada para modificação e revisão. São gerados vários ciclos, no desenvolvimento por exemplo podemos identificar a fase de análise, projeto, teste e implementação, onde os ciclos são considerados um subconjunto de requisitos, que são desenvolvidos e alocados a um ciclo de desenvolvimento, o próximo ciclo o novo conjunto de requisitos que será desenvolvido, produzira um novo incremento que contém extensões e refinamentos do ciclo anterior, sendo assim o desenvolvimento vai evoluindo, até que o sistema esteja completo.

* 1. **Casos de Uso**

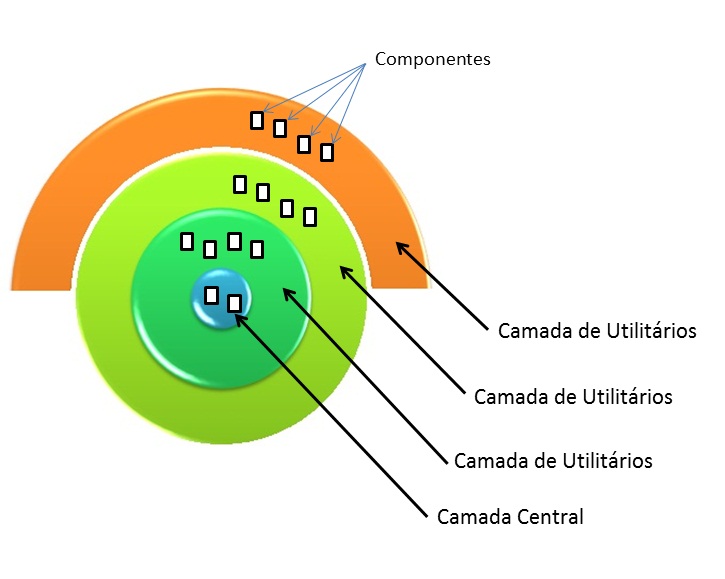
Caso de uso é uma técnica de modelagem com a finalidade de descrever os requisitos funcionais do sistema de forma clara e consistente.

Segundo Ivar Jacobson, podemos dizer que um caso de uso é um “documento narrativo que descreve a sequência de eventos de um ator que usa um sistema para completar um processo”

Os Diagramas de Casos de Uso são representações dos mesmos e podem ser representados por uma elipse contendo, internamente, o nome do caso de uso. Esta elipse pode estar ligada por um traço a um boneco, onde o boneco indica o ator e o traço indica a interação entre eles.

* 1. **Arquitetura em Três Camadas (3 TIER)**

A estrutura básica de uma arquitetura em camadas estará ilustrada na figura a baixo. Um certo número de camadas diferentes é definido, cada uma realizando operações que se tornam progressivamente mais próximas do conjunto de instruções de máquina. Na camada exterior; os componentes servem as operações da interface do usuário. Na camada mais interna os componentes realizam a interface com o sistema operacional. As camadas intermediarias fornecem serviços utilitários e funções do software de aplicação.



* 1. **Microsoft Visual Studio**

Conjunto de ferramentas utilizado para desenvolver aplicativos web e desktop. Foi desenvolvida pela *Microsoft*, e foi dedicado ao *framework.net* e *Asp.Net* (plataforma web), onde pode ser desenvolvido aplicativos, sites e serviços para esta plataforma.

* 1. **C# (C Sharp)**

*C#* lê- se como “cê chárp” no português e “si sharp” para o inglês. Foi criado pela Microsoft para a plataforma .NET, é uma linguagem orientada a objetos que foi baseado nas linguagens C++ e Java, C# é considerada uma linguagem símbolo de .Net pois, criaram do zero sem preocupação de com o código legado, desenvolvida para rodar em uma plataforma nova e foi criado primeiro o seu copilador. A linguagem foi desenvolvida por vários programadores, mas o principal foi Anders Hejlsberg, que é o criador de Turbo Pascal e Delphi.

Na linguagem não é inclusa nenhuma biblioteca de classes ou funções, a mesma está ligada ao framework.Net onde obtém suas classes e funções de execução. Ao desenvolver na linguem percebesse que é organizado por conjuntos de namespaces, exemplo *System.Collections (estruturas de dados) e System.Drawing (para gráficos)* e suporta encapsulamento, polimorfismo e herança.

C# foi submetida a ECMA, para padronização formal, onde em 2001 foi liberada a especificação ECMA - 334, e em 2003 tornou padrão ISSO (ISO\IEC 23270). A Microsoft continua induzindo mudanças nas linguagens, onde C# faz parte de sua estratégia de software.

* 1. **Designer Patterns**

Designer Pattern (Padrões de Projetos), onde propõe uma solução para determinados problemas que ocorrem com frequência. Os padrões de projeto visam facilitar a reutilização de soluções de projetos.

Os designer patterns tiveram suas raízes no trabalho do engenheiro civil Chistopher Alexander, onde estabeleceu padrões como nome e contexto, por exemplo. Neste trabalho Chistopher descreveu como resolver problemas baseados em suas próprias experiências. Segundo Chistopher “cada *pattern descreve um problema que ocorre com frequência em nosso ambiente, e então descreve o núcleo da solução para tal problema, de uma maneira que possamos utilizar esta solução milhares de vezes*.”

Baseados nos conceitos introduzidos por Chistopher, em 1995 um grupo de quatro programadores ficaram conhecidos como a gangue dos quatro (*Gang of Four – Gof*) ao publicar o livro “*Designer Patterns: Elements of Reusable Object-Oriented Software*”. Eram eles: Erich Gamma, Richard Helm, Ralph Johnson e John Vlissides. O livro trazia 23 padrões que foram organizados em 03 famílias:

* Padrões de criação: relacionados a criação de objetos;
* Padrões estruturais: tratam das associações entre classes e objetos;
* Padrões comportamentais: tratam das interações e divisões de responsabilidades entre as classes ou objetos.

* 1. **Singleton**

Considerado um padrão criacional, o *Singleton* garante que apenas uma instância do objeto seja instância em todo o projeto, além de ter um ponto de acesso global a essa instância.

Outra funcionalidade é quando precisamos de uma classe de *Log* para todo o sistema, sem precisar instanciar a classe, utilizando os recursos do sistema, pois não é preciso chamar o construtor para a criação da instância, onde seu acesso é restrito (método privado).

Para criar a instancia da classe usa o método público, onde o mesmo vai fazer o controle da instancia, permitindo que ela só possa ser feita uma vez.

* 1. **Windows Presentation Foundation (WPF)**

Tecnologia criada pela Microsoft para desenvolver aplicativos e interfaces, não depende de plataforma, sendo assim o código-fonte funciona para web e desktop. O programa que utiliza o Wpf tem duas partes, um que utiliza o arquivo XML que contém uma característica chamada XAML (eXtended Aplication Markup Language, pronuncia-se “zémel”) e um código .Net.

Suas principais características são:

º Flexibilidade na interface, independente do código;

º Utilizada em desktop e web;

º Utiliza XAML e código. Net

º Melhora a interface do hardware do usuário através do sistema operacional

* 1. **Model View Controller (MVC)**

O *Mvc* é composto pelo modelo, visão e controlador, sendo que no começo servia para mapear o processo de entrada, processamento e saída, assim era possível ter os três processos mapeados.

*Controller* é responsável por controlar as entradas do mouse e teclado, mapear as ações e enviar para o Model ou para a *View*, onde pode ser realizada a alteração, o Model também pode gerenciar vários dados, sendo assim o mesmo responde o estado e como mudar o estado. A *View* fica responsável para apresentar ao usuário as informações que são geradas por gráficos e textos, sendo assim ela apenas recebe as informações do *Model* e *Controller* e exibe.

* 1. **Object-Relational Mapping (ORM)**

O *Orm* é um framework ou conjunto de classes que facilita o do desenvolvedor, pois não precisa escrever códigos a todo o momento para conexão com bancos, *querys* de *Sql*, mesmo assim as características da orientação ao objeto são preservadas.

Facilita a manutenção, o código fica mais limpo e se obtém uma ótima performance da padronização.

* 1. **.Net Frameworks**

O *.Net Frameworks* é considerado um ambiente de tempo de execução, gerencia aplicativos que utilizam o *.Net Framework.*

Consistem em uma linguagem comum, é fornecido gerenciamento de memória, administra uma imensa biblioteca de classes, o que permite os programadores aproveitar ao máximo o código.

* 1. **Sql Server Express 2012**

Programa de gerenciador de banco de dados, é da Microsoft que foi criado a partir de uma parceria com a Sybase em 1988, onde inicialmente era um complementar do Windows NT, com o decorrer do tempo, foi sendo aperfeiçoado e é um sistema robusto e usado por diversas corporativas e comporta diversas plataformas. O *Sql* (Structured Query Language) está sendo um dos mais usados do mundo, competindo fortemente com o *Mysql* e *Oracle*.

O backup exige menos do I\O (entrada\saída), e diminui os custos para manter online, existe a computação de bloqueio que o usuário pode aumentar a profundidade de hierarquia. Umas das principais características é que a linguagem trabalha com conjuntos de registros de dados, enquanto as outros, foram feitas para tratar um registro de cada vez, pois o *Sql* não é procedural sendo assim não precisa detalhar a tarefa que precisa, a linguagem descreve o que o usuário final deseja.

A linguem se divide em duas: DDL (linguagem de Definição de Dados) - Usa instruções para descrever o esquema das tabelas, e a DML (linguagem de Manipulação de Dados) - Usa instruções para manipular os dados.

* 1. **Testes**

Um software testado, revela se as funcionalidades realmente estão de acordo com o que foi especificado no projeto e apontam eventuais erros para serem corrigidos.

Pressman reforça que

*Inicialmente, a engenharia de sistemas define o papel do software a*

*passar a análise dos requisitos de software, onde são estabelecidos o*

*domínio da informação, função, comportamento, desempenho, restrições*

*e critérios de validação para o software. Deslocando-se para o interior da*

*espiral, chega-se ao projeto e finalmente á codificação. Para desenvolver*

*software de computador, percorre-se a espiral para o interior (sentido anti-horario)*

*ao longo de linhas que indicam a diminuição do nível de abstração*

*em cada volta.*

Uma estratégia de testes é definida pelos seguintes testes:

* Teste de Unidade

É realizado nas unidades do sistema para garantir a integridade, funcionalidades e comportamento das mesmas. Geralmente são realizados pelos desenvolvedores.

* Teste de Integração

São testes que focam a arquitetura de software e verifica a integridade entre os componentes.

* Teste de Validação

Foram projetados para validar os requisitos da modelagem.

* Teste de Sistema

São testes que verificam o sistema como um todo. Este teste abrange a combinação entre software e hardware, base de dados e usuários e é analisado como o sistema se comporta.

1. **ORGANIZAÇÃO CLIENTE**

A organização cliente cujo nome da personalização é Hermes Bar, através da sua qualificação, é pessoa jurídica de direito privado. Sua trajetória sempre esteve ligada às manifestações músico-culturais, situado à rua Avenida Iguaçu nº 2504, bairro Água Verde, fone 41 3018 9320, CEP 80240-030 no município de Curitiba - PR.

O comércio possui um histórico de 56 (cinquenta e seis) anos de existência e neste período presta um atendimento diferenciado no quesito música de boa qualidade.

Com o aumento da popularidade do estabelecimento e quantidade de clientes que o frequenta, faz-se necessário uma aplicação que auxilie no controle e gerenciamento do negócio como um todo. Desta forma, nossa proposta visa colaborar com o Hermes Bar no gerenciamento do seu negócio.

1. **DIAGNOSTICO DO AMBIENTE**

Atualmente o estabelecimento não conta com nenhum tipo de processo informatizado, sendo assim, todas as atividades administrativas feitas manualmente.

Orçamentos, vendas, controle de estoque são realizados manualmente e controlados somente utilizando as fichas de consumo.

O controle de estoque é realizado visualmente. O cliente verifica os itens que estão em menor quantidade e providencia sua reposição junto aos fornecedores.

Não há cadastro de clientes nem de fornecedores.

Diante deste cenário, o controle de estoque e gerenciamento das rotinas administrativas ficam extremamente dependentes da memória e percepção do cliente, caso o mesmo esqueça de acionar algum fornecedor para compra, o estabelecimento fica sem o produto, ou ainda, perca alguma ficha de consumo utilizada para controle, poderá ocasionar um grande problema financeiro.

1. **OBJETIVOS**

Esse projeto tem como objetivo a intervenção e criação de um sistema que colabore com a empresa, automatizando todos os processos internos, hoje realizados manualmente e sem controle.

A aplicação deve facilitar o controle de gestão, a manutenção do estoque, desde a entrada do cliente á saída de um produto, para que os processos sejam mais confiáveis.

Temos o objetivo de auxiliar o cliente nas suas atividades internas no estabelecimento, onde o sistema trará benefícios como controle de estoque, entrada dos clientes no bar e restaurante, os pedidos enviados para a cozinha e o controle de comanda.

Nossa proposta é um sistema composto por dois módulos, sendo eles:

* Módulo gestão
* Módulo pedido

Acredita-se que a solução atenda as necessidades do cliente.

1. **DESENVOLVIMENTO**

O desenvolvimento do sistema Hermes Management Assistant teve início no mês de novembro de 2014, juntamente com a escolha da organização cliente. Foi elaborado um cronograma e definido uma metodologia de gerenciamento de projetos pois atuaremos sobre um objetivo proposto e um padrão de qualidade estabelecido, considerando custos definidos previamente.

Após a escolha da organização cliente, realizamos uma entrevista com a representante Leticia Sanches Dutra, onde foram coletadas as necessidades iniciais que o sistema deveria possuir. Foi definido que o sistema possuiria 02 (dois) módulos, sendo eles Gestão e Pedidos, que serão desenvolvidos na plataforma *Microsoft Windows* com linguagem de programação C# e também foi definido que o banco de dados seria *SQL Server Express* com a versão 12. Baseados nessas informações foi elaborado o Escopo do Produto.

Com a definição do escopo do produto, o próximo passo foi a definição do escopo do projeto. Para isso foi criado o EAP que nos dá uma visão do projeto como um todo.

Após a conclusão desses processos, foram definidas as partes interessadas (*stakeholders*), assim como o cliente. Foram coletados requisitos funcionais necessários, realizado um levantamento das regras de negócio, assim como as necessidades e funcionalidades que o sistema deverá atender. Foi definido que o sistema terá um usuário administrador onde terá acesso limitado a todos os módulos do sistema. Essas informações coletadas formaram o Documento de Visão.

Conforme a metodologia de Gerenciamento de Projetos, foi estabelecido um planejamento de comunicação entre as partes interessadas, onde focamos em estabelecer a estratégia utilizadas para este fim. Foi definido que esta comunicação seria realizada através de reuniões presenciais e correio eletrônico.

De forma a otimizar a comunicação entre a equipe de desenvolvimento, foi adotadoum recurso de teleconferência denominado *Skype,* uma ferramentaque permite comunicação através da internet com o uso de voz sobre IP, além de reuniões semanais realizados no Centro Tecnológico da Universidade Positivo. Foi adotado também como método de desenvolvimento o Interativo Incremental, onde define que quando uma determinada parte do projeto está desenvolvida, apresentamos para o cliente.

Seguindo o escopo do projeto definido através do EAP, foi necessário estipular um Plano de Gerenciamento de Testes, no qual foi definido os integrantes responsáveis para garantir a qualidade e periodicidade do sistema.

Os requisitos funcionais do sistema, após registrados no documento de visão precisam ser representados. Isso é possível através da modelagem dos Casos de Uso. O Hermes Assistant Menagement inicialmente possui vários casos de uso, onde pode-se ter uma visão macro do projeto através do Diagrama de Caso de Uso.

Após a criação dos Casos de Uso, foram definidas as classes da aplicação e seus atributos, que estão representados através do Diagrama de Classes.

O desenvolvimento do código-fonte do Hermes Assistant Menagement teve início dia 21 de novembro, onde foi utilizado todas as informações presentes neste documento.

Como parte do código-fonte, é possível consultar a *bll* (Business Layer Logic) onde encontra-se a camada lógica de negócio, o core do sistema e onde estão todas as regras de negócio. Esta camada realiza todas as validações existentes no e é nela que conseguimos verificar se os dados foram gravados ou não no banco. A classe *dao* (Data Access Object) é utilizada para realizar o acesso ao banco de dados, é a camada que realiza o select, update e insert, por exemplo. Temos a classe *view* para a realização de interface, onde a mesma chama a classe *bll* e repassa os dados. Utilizamos a camada *útil* onde esta camada possui dados que todas as outras camadas poderão usar. E por fim, a camada Model, onde fica apenas o modelo e os objetos.

A medida que o código-fonte foi desenvolvido, foram realizados testes de unidade e regressão para garantir a qualidade do mesmo. A vantagem de realizar testes de unidade é que a medida que a equipe de desenvolvimento termina um módulo, ele é testado e quando encontrado erros, ficam mais fáceis de identificar e corrigi-los e quando a equipe termina o desenvolvimento de outro módulo ou realiza alguma correção, todos os módulos são testados novamente para garantir a qualidade do software.

1. **CONSIDERAÇÕES FINAIS**

Ao decorrer do projeto, desde de sua elaboração, planejamento e desenvolvimento, a equipe passou por várias fases, como: a escolha do projeto, escolha do cliente, se seria deferido o projeto pela instituição, o aceite do orientador, mas a maior expectativa da equipe era ver o nosso sistema funcionando, uma realização para a equipe, em poder resolver um problema dentro da empresa, aplicando todo o conhecimento adquirido no decorrer do curso, adquirir conhecimentos fora do curso, o trabalho em equipe, onde todos se empenharam para a elaboração e execução do projeto.

Durante o projeto ouve momentos de discordância de ideias, que serviram para o grupo aprender e exercitar valores de compreensão e paciência, a saber lidar com opiniões diferentes, aplicar o conhecimento profissional que cada um possui, dentro do projeto do *PAP,* onde todos estavam motivados na concretização do sistema.

Podemos dizer, que foi um dos maiores aprendizados que adquirimos, ganhamos experiência, se fortalecemos e acima de tudo colocamos em pratica, tudo o que foi nos ensinados.

**REFERÊNCIAS BIBLIOGRÁFICAS**

GAMMA, **Erich Design Patterns: Elements of Reusable Object-Oriented Software** Addison Wesley, 1995.

PRESSMAN, Roger S**. Engenharia de software**. 7. edição. Editora MCGrawHill: Porto Alefre, 2010.

PMI **Um Guia do Conjunto de Conhecimentos em Gerenciamento de Projetos - Quarta edição (Guia PMBOK**), 2008.

SOMMERVILLE, **Engenharia de Software, 8ª edição** Pearson Addison-Wesley, 2007.

ABNT NBR **ISO 10006 Gestão da qualidade** – Diretrizes para a qualidade no gerenciamento de Projetos ABNT, Dez 2000

PMI **Um Guia do Conjunto de Conhecimentos em Gerenciamento de Projetos Terceira edição (Guia PMBOK)**,2004.

**Sites Pesquisados:**

QUALIDADE, Comitê Brasileiro da. **Diretrizes para a qualidade no gerenciamento de projetos**. Disponível em:

<http://licenciadorambiental.com.br/wp-content/uploads/2015/01/NBR-10.006-Gest%C3%A3o-da-Qualidade-diretrizes-para-a-qualidade-no-gerenciamento-de-projetos.pdf>

Acesso em: 05/05/2015.

JAVAFREE.ORG. **Designer Pattern**. Disponível em:

<http://javafree.uol.com.br/wiki/Design%20Pattern>

Acesso em: 05/05/2015.

LINHADECÓDIGO. **Introdução a Design Patterns.** Disponível em: <http://www.linhadecodigo.com.br/artigo/345/introducao-a-design-patterns.aspx>

Acesso em: 05/05/2015.

CODEPLEX **Welcome to Windows Presentation Foundation**! Disponível

em:<http://wpf.codeplex.com/>

Acesso em: 13/03/2015

CODEPLEX **WPF Toolkit – January 2009 Release** Disponível em:

<http://wpf.codeplex.com/releases/view/22567>

<https://msdn.microsoft.com/pt-br/library/cc564903.aspx>

Acesso em: 10/03/2015.

Git **Primeiros Passos** Disponível em: <http://git-scm.com/book/pt-br/v1/Primeiros-passos-No%C3%A7%C3%B5es-B%C3%A1sicas-de-Git> Acessado em: 10/03/2015

**Kanban Gestão** Disponível em:

<http://www.knoow.net/cienceconempr/gestao/kanban.htm>\ <http://www.significados.com.br/kanban/>

Acesso em 10/03/2015

**Visual Studio** Disponível em:

<https://msdn.microsoft.com/pt-br/vstudio/ff431702>

Acesso 13/03/2015

Iterativo

<http://www.linhadecodigo.com.br/artigo/2108/evolucao-da-metodologia-do-desenvolvimento-de-sistemas.aspx>

Acesso 13/03/2015

MSDN I**ntrodução ao Visual C#** Disponível em:

<http://msdn.microsoft.com/pt-br/vcsharp/dd919145.aspx>

Acesso em: 13/03/2015

**Mvc** Disponível em:

<http://www.devmedia.com.br/introducao-ao-padrao-mvc/29308>

Acesso em 13/03/2015

**Orm** Disponível em:

<http://www.devmedia.com.br/analise-dos-melhores-orm-object-relational-mapping-para-plataforma-net/5548>

Acesso em 13/03/2015