Aplicativo de Desdobramento das Funções da Qualidade (QFD) Utilizando Conceitos de Programação Orientada a Objetos

App for Quality Function Deployment (QFD) Using Concepts of Object-Oriented Programming

PINHEIRO, Rogélio Carpes; Mestre em Design & Tecnologia; UFRGS

rogelio.pinheiro@gmail.com

VIARO, Felipe Schneider; Mestre em Design & Tecnologia; UFRGS

felipesviaro@gmail.com

TEIXEIRA, Fábio Gonçalves; Pós-Doutor em Engenharia Mecânica; UFRGS

fabiogt@ufrgs.br

SILVA, Régio Pierre da; Doutor em Engenharia de Produção; UFRGS

regio@ufrgs.br

Resumo

O Desdobramento das Funções da Qualidade (QFD) é utilizado em diversas indústrias na gestão da qualidade e no desenvolvimento de novos produtos. Planilhas eletrônicas e *softwares* pagos são as ferramentas mais utilizadas para esse propósito e apresentam limitações de usabilidade e custo. Tendências atuais apontam para o uso de dispositivos *mobile* tanto em atividades de ensino e aprendizagem quanto no ambiente de trabalho. Portanto, o objetivo deste trabalho é desenvolver um aplicativo *desktop* e *mobile* de QFD utilizando métodos da Programação Orientada a Objetos (POO). O método de desenvolvimento engloba análises, programação e testes com protótipos. O aplicativo desenvolvido proporciona fácil utilização em um ambiente multiplataforma e de código aberto. Esta proposta estimula o projeto da qualidade de produtos, que, através de uma ferramenta prática e flexível, aumenta o potencial de aplicação do QFD no processo projetual.

Palavras Chave: Desdobramento das Funções da Qualidade (QFD), Processo de Desenvolvimento de Produtos (PDP), Programação orientada a objetos (POO).

Abstract

Quality Function Deployment (QFD) is used in various industries in quality management and in the development of new products. Spreadsheets and paid software are the most used tools for this purpose and have limitations on usability and cost. Current trends point to the use of mobile devices in teaching and learning activities as well as in the work environment. Therefore, the objective of this work is to develop a desktop and mobile QFD application using Object Oriented Programming (OOP) methods. The development method encompasses analysis, programming and testing with prototypes. The developed application provides easy use in a cross-platform and open source environment. This proposal stimulates product quality design, which, through a practical and flexible tool, increases the potential for applying QFD in the design process.

Keywords: *Quality Function Deployment (QFD), Product Development Process (PDP), Object-Oriented Programming (OOP).*

1 Introdução

A capacidade de gerar boas ideias é uma das principais características compartilhadas por designers e equipes de design bem-sucedidas no desenvolvimento de produtos. Nesse sentido, a quantidade é um fator importante, pois são necessárias aproximadamente 100 ideias iniciais para se chegar a um produto de sucesso (BACK *et. al*, 2008).

Ao longo do processo projetual, as melhores ideias vão sendo sistematicamente refinadas em conceitos e concepções, os quais são testados e revisados, para que, finalmente, seja escolhida e detalhada a melhor opção, a qual será posteriormente produzida e distribuída. A execução desse processo sistemático, chamado de Processo de Desenvolvimento de Produtos (PDP), é uma maneira de impulsionar a qualidade técnica do mesmo e atender as metas estabelecidas durante a fase de seu planejamento (BACK *et al.*, 2008; BAXTER, 2010).

Fatores como a competitividade dos mercados e o alto nível de cobrança em relação à assertividade e criatividade de designers levaram profissionais e teóricos a desenvolverem métodos e ferramentas adicionais para contribuir com a eficiência do processo projetual (BOMFIM, 1995; LÖBACH, 2001). Nesse caso, pode-se apontar o Modelo Kano e o método QFD (acrônimo de *Quality Function Deployment*) como meios de contribuir para a qualidade final do produto. O primeiro orienta o designer a atingir a qualidade através do atendimento das necessidades básicas e desejos dos usuários, assim como, da inclusão de fatores de excitação do produto que geram surpresas positivas nos usuários (BAXTER, 2010).

O QFD, por sua vez, constitui um método completo para o projeto da qualidade através do desdobramento sistemático da ferramenta casa da qualidade, que pode ser aplicada de forma única ou em aplicações sucessivas nas macrofases de projeto, fabricação e pós-venda (AKAO, 1990; BAXTER, 2010). Um dos principais usos desse método está no estabelecimento das relações entre os requisitos do usuário[1] e requisitos de projeto, o que permite hierarquizar as características de maior influência na qualidade do produto. Essa flexibilidade e sistematização tornam o QFD um método com grande potencial de aplicação em diferentes contextos, desde salas de aula até ambientes profissionais (CHAN; WU, 2002).

A convergência tecnológica que é verificada atualmente, tendo como importantes vetores o acesso universal à internet de alta velocidade, bem como um mercado com dispositivos móveis com alta capacidade de processamento e custos acessíveis, possibilita que computadores potentes e interconectados estejam nas mãos de profissionais e acadêmicos, virtualmente em qualquer lugar. Os dispositivos móveis já ultrapassam os computadores de mesa no acesso à internet e, de maneira geral, atividades profissionais ganham cada vez mais espaço em celulares e *tablets* (BOSOMWORTH, 2015).

A recente Reforma Trabalhista aprovada em 2017 pelo Governo Brasileiro, a qual regularizou o *home office* como parte integrante da jornada de trabalho, também aponta para a

[1] Alguns autores utilizam termos distintos para fazer referência ao usuário final do produto. Back *et. al* (2008) utilizam o termo *usuário* para se referir ao usuário final e, *cliente*, para quem contrata os serviços de design. Por outro lado, Rozenfeld *et. al* (2008) utilizam o termo *cliente* para se referir ao consumidor e usuário final. Neste trabalho, tanto cliente como usuário possuem a mesma conotação - o usuário final do produto.

mobilidade como uma tendência (BRASIL, 2018). Segundo Andriolli (2018), 84% dos profissionais utilizam dispositivos pessoais em ambientes de trabalho. Na educação, existe uma expectativa da sociedade e dos alunos para que as novas tecnologias sejam integradas à sala de aula (NETTO, 2005), sendo as possibilidades de integração e aproveitamento múltiplas (LÉVY, 2010). Assim, ambientes acadêmicos e profissionais devem estar integrados no ciberespaço, o que significa incluir as ferramentas digitais nas rotinas e processos de profissionais e estudantes, sendo as plataformas *mobile* as que apresentam maior potencial de utilização.

O uso de *softwares* para a criação de QFDs aumenta o potencial produtivo das equipes de projeto, uma vez que a ferramenta digital permite que os membros da equipe se concentrem nas atividades essenciais do método, deixando de lado tarefas mais burocráticas como a documentação e a execução de cálculos (HERZWURM; SCHOCKERT; MELLIS, 1997). Observa-se que em sua grande maioria esses *softwares* são pagos e, portanto, menos acessíveis a um público representativo de estudantes, profissionais autônomos e Micro e Pequenas Empresas (MPEs). Dessa forma, grande parte das empresas e universidades acabam por utilizar planilhas eletrônicas na execução de QFDs, através da criação de modelos próprios ou da adaptação de *templates* encontrados na internet. Apesar de tais planilhas apresentarem um funcionamento básico razoável, geralmente há limitações técnicas, considerando que não foram especificamente programadas para esse fim.

Considerando os benefícios do uso de ferramentas digitais no PDP, em particular o QFD como meio de promover a qualidade do produto, o objetivo deste trabalho é automatizar e tornar acessível o método QFD através da criação de um aplicativo digital de código aberto com possibilidades de uso em ambientes *desktop* e *mobile*. O artigo está estruturado da seguinte forma: o método QFD e a Programação Orientada a Objetos (POO) estabelecem o quadro teórico pertinente; apresenta-se o processo de desenvolvimento do aplicativo, o qual engloba definições de suas funcionalidades, análise de similares, programação do aplicativo e testes de uso; e, por fim, são apresentadas as principais conclusões e sugestões de trabalhos futuros.

2 Desdobramento das Funções da Qualidade (QFD)

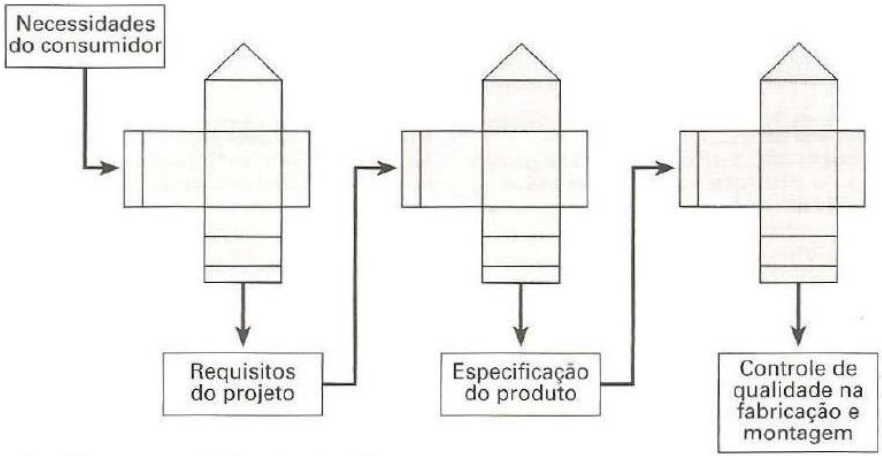
Pelo ponto de vista histórico, a qualidade passou por quatro fases de desenvolvimento: inspeção, supervisão de processo, controle e gerenciamento estratégico de qualidade. Com a revolução das filosofias de manufatura entre o final da década de 1970 e início da década de 1980, o Desdobramento das Funções da Qualidade (QFD), foi desenvolvido pela escola japonesa de manufatura, com o intuito de propiciar redução de custos e melhoria nos métodos de fabricação (PRASAD, 1998; ROZENFELD *et al.,* 2006).

Inicialmente utilizada pela Mitsubishi Heavy Industries e seguida por inúmeras grandes empresas japonesas, esse método chegou ao ocidente na segunda metade da década de 1980. A sua importância pode ser constatada através de companhias renomadas que o utilizam, como Ford, GM, Chrysler, AT&T, HP, entre outras. Com a crescente aceitação do QFD nas áreas de qualidade, ele também passou a ser utilizado no desenvolvimento de produtos, sendo muitas vezes combinado com outras técnicas de seleção, como a matriz de Pugh e o método TRIZ (PRASAD, 1998).

O uso do QFD no desenvolvimento de produtos possibilita o estabelecimento de relações entre necessidades do cliente e requisitos de projeto, análise de concorrentes, especificação de metas e avaliação dos conflitos entre os requisitos de projeto com as respectivas dificuldades

técnicas (ROZENFELD *et al.,* 2006). O QFD realizado no pré-desenvolvimento do produto passa por desdobramentos sistemáticos para assegurar que os requisitos iniciais sejam atingidos em todas as fases do ciclo de vida do produto. Por exemplo, o QFD inicial pode gerar outros para subsistemas, processos de fabricação e pós venda (Figura 1). Idealmente, esses desdobramentos iniciam-se com cada requisito e se estendem para todos os componentes ou processos, e a qualidade global do produto é formada através dessa trama de relações (AKAO, 1990).

Figura 1 - Desdobramentos sistemáticos de QFDs



Fonte: Baxter (2010)

Além de sua aplicação prática como ferramenta de transformação e refinamento de informações, a síntese visual obtida através do uso do QFD adiciona uma funcionalidade adicional como ferramenta de comunicação. Com a ferramenta devidamente preenchida, a equipe de projetistas ganha uma evidência física que serve como base para discussões sobre a importância dos requisitos de projeto, sobre as metas estabelecidas e outros parâmetros que determinarão o futuro do produto em desenvolvimento (MORRIS, 2009).

3 Programação Orientada a Objetos

Muitas linguagens de programação modernas suportam alguma forma de Programação Orientada a Objetos, comumente abreviada como POO, que pode ser definida como uma linguagem de programação básica, em que os objetos têm seus próprios atributos e métodos, formando classes hierarquicamente organizadas. De acordo com essa definição, o objeto é um modelo (abstração) de um código original (BERDONOSOV; ZHIVOTOVAB; SYCHEVA, 2015).

Nesse conceito, a classe é uma declaração abstrata de atributos e métodos para um grupo de objetos semelhantes, que são chamados de instâncias. O atributo é um parâmetro declarado que caracteriza o objeto (instância da classe). O método, por sua vez, é declarado em um procedimento que define o comportamento de instâncias de classe.

A maioria das linguagens POO se baseia em três conceitos fundamentais (CANTÙ, 2016):

- **Classes:** tipos de dados com uma interface pública e uma estrutura de dados privada, implementando encapsulamento. As instâncias desses tipos de dados geralmente são chamadas de objetos.
- **Extensibilidade ou herança de classe:** capacidade de estender um tipo de dado com novos

recursos sem modificar o original.

- **Polimorfismo ou ligação tardia:** capacidade de se referir a objetos de diferentes classes com uma interface uniforme e ainda operar em objetos da maneira definida pelo seu tipo específico.

Outra definição de POO é proposta por Budd (1997), que define como quatro seus mecanismos principais:

- **Abstração:** processo de identificação das características essenciais de um objeto que o distingue de todos os outros e, portanto, fornece limites conceituais definidos.
- **Encapsulamento:** método de compartimentação dos elementos de uma abstração que constitui sua estrutura e comportamento.
- **Polimorfismo:** capacidade de poder atribuir um significado ou uso distinto em diferentes contextos.
- **Herança:** mecanismo para declarar novos tipos de dados com base em tipos existentes, de tal forma que os atributos e métodos dos dados iniciais tornam-se membros do subtipo.

Observa-se que as definições de POO de Budd e Cantù se encaixam em uma mesma forma de pensar a linguagem orientada ao objeto, complementando-se uma à outra.

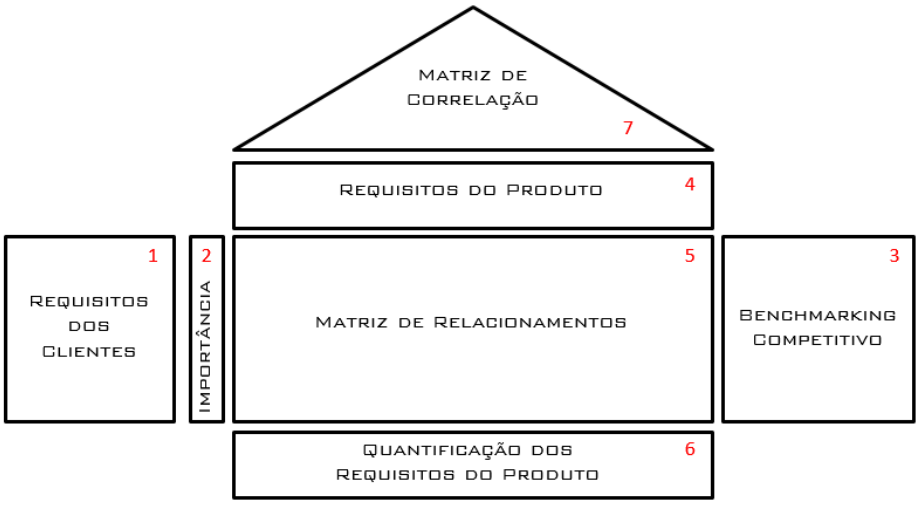
4 Desenvolvimento do Aplicativo QFD

O aplicativo QFD proposto neste trabalho foi desenvolvido segundo o seguinte processo: i) análise da ferramenta a fim de estabelecer as funcionalidades do aplicativo; ii) análise de similares para extrair requisitos de projeto; iii) programação; iv) testes com o protótipo.

4.1 Análise do QFD no Contexto de Desenvolvimento de Produtos

Existem diversas variações do QFD, sendo a proposta de Rozenfeld *et al.* (2006) uma das mais utilizadas no desenvolvimento de produtos. O mesmo consiste em sete áreas fundamentais que, dependendo do nível de detalhamento requerido, podem-se incluir ou excluir campos ou subcomponentes específicos (Figura 2).

Figura 2 - Modelo usual de QFD



Fonte: Rozenfeld *et al.* (2006)

Cada um dos sete campos do QFD são detalhados a seguir, conforme Sivaloganathan e Evbuomwan (1997):

1) **Requisitos dos Clientes:** Os consumidores são inicialmente ouvidos para a geração da lista de requisitos do cliente, que compreendem as necessidades e expectativas dos usuários. Em algumas publicações, a voz do consumidor é traduzida como requisitos de mercado.

2) **Importância**: Para cada requisito é elencado um valor de importância, que normalmente utiliza a escala de 1 a 5 ou 1 a 10. Esse valor é definido pelo cliente, e após a soma, é feita uma valoração em porcentagem, para se entender quais requisitos têm peso maior, ou seja, que devem ser prioritariamente atingidos.

3) ***Benchmarking* Competitivo:** Essa categoria descreve a importância relativa dos produtos concorrentes na visão dos clientes, mostrando, dessa forma, como satisfazer seus requisitos de uma maneira priorizada. São elencados normalmente com o nome de competidores diretos ou segmentos de mercado.

4) **Requisitos do Produto:** Os fabricantes definem essa área como um conjunto de características de qualidade através das quais as demandas dos clientes podem ser atingidas. Considerando essas prioridades, a empresa pode medir e controlar a qualidade para assegurar que os requisitos do cliente sejam atendidos.

   Como exemplo, se o requisito do usuário para um veículo for "boa dirigibilidade", como se pode atingir isso? É possível listar vários recursos para atender essa demanda, como, por exemplo: Amortecedores, Estabilidade Mecânica, Controle Eletrônico de Estabilidade, Centro de Gravidade e de Massa, Peso, Materiais, etc. Assim, para cada Requisito do Cliente, existirá pelo menos um Requisito de Projeto.

5) **Matriz de Relacionamentos:** As matrizes de relações podem utilizar números ou símbolos, dependendo do contexto e do propósito a partir do qual o QFD está sendo realizado. As escalas numéricas utilizadas normalmente são 1, 3 e 5 ou 1, 3 e 9, sendo 1 uma relação fraca, 3 média e 5 ou 9, uma relação forte. A escala de 1 a 5 representa uma progressão aritmética, já a escala de 1 a 9 simboliza uma progressão geométrica. Assim, a última tende a discriminar as relações fracas de forma drástica contra as relações fortes, enquanto a primeira discrimina ambas de igual maneira.

6) **Quantificação dos Requisitos do Produto:** Consiste na lista que identifica a viabilidade e compila os valores estipulados para cada um dos Requisitos de Projeto. Em outras palavras, para cada COMO, existe um valor correspondente para um valor de QUANTO. Nessa área, todos os parâmetros devem ser quantificáveis através de porcentagens ou valores limites (metas), utilizando uma unidade de medida adequada.

7) **Matriz de Correlações:** A matriz de correlação, ou telhado da casa da qualidade, verifica a intensidade do relacionamento entre os Requisitos de Projeto. Isso torna possível verificar o quanto um requisito de projeto influencia outro, podendo, inclusive, apontar conflitos existentes e auxiliar os projetistas a tomar decisões conscientes. Assim como na Matriz de Relacionamentos, utilizam-se números ou símbolos para representar o grau de relações.

4.2 Análise de Similares

Foram analisados *softwares* e aplicativos QFD a partir de critérios relevantes para o desenvolvimento do aplicativo aqui proposto. A ideia foi obter uma vasta cobertura de plataformas e variação de preços a fim de estabelecer uma proposta acessível e com boa usabilidade. A análise ocorreu a partir do uso das ferramentas digitais, observando-se funcionalidades, interface, custos e potencial de uso em plataformas variadas. A fim de possibilitar a análise dos aplicativos, foi aplicado o exemplo do produto cafezinho[2]. Os resultados dessa análise são apresentados no Quadro 1.

Quadro 1 – Comparativo entre aplicativos QFD

App	**Custo**	**Plataforma**	**Interface**	**Funcionalidades**
QFD Capture	US$ 500 para empresas; US$ 250 para universidades	*Desktop*	Aparência similar às planilhas do Excel, o que dificulta seu uso *mobile*.	Pode criar QFDs com campos predeterminados, podendo assim ter mais ou menos características e interrelações.
QFD House of Quality	US$ 2,99	iOS	Simples, com falta de contraste entre os campos; Dimensões e fontes adequadas a *tablets*.	Necessidade de especificar previamente a quantidade de requisitos, competidores e dimensões das células, sem opção de alteração posterior; sem cálculos de peso relativo ou importância dos requisitos.
House of Quality App	Sem custo	*Desktop web-based*	Interface simples, com botões na parte superior para edição de requisitos e concorrentes. As demais interações são feitas diretamente dentro das matrizes.	Campos adicionais de Direção de Melhora e Níveis de Dificuldade para atingir requisitos de produto; Gráfico colorido na análise dos concorrentes; ênfase de colunas e linhas relacionadas na matriz de correlações, em eventos *On Mouse Over*; compartilhamento do QFD com outros usuários através do Google Drive; não permite edição síncrona.
Templates de Excel	Sem custo, em sua grande maioria	*Desktop; Mobile*	Menus padrões do programa; planilha varia de acordo com o *template*	Os diferentes *templates* existentes na internet permitem a criação de QFDs de diferentes complexidades; Engloba a facilidade e familiaridade de uso do Excel, juntamente com suas limitações.

Fonte: autores

[2] **Passo a passo para construção da casa da qualidade** (2017).

Com a análise e teste desses aplicativos de QFD, foi possível gerar diretrizes para a criação do aplicativo proposto:

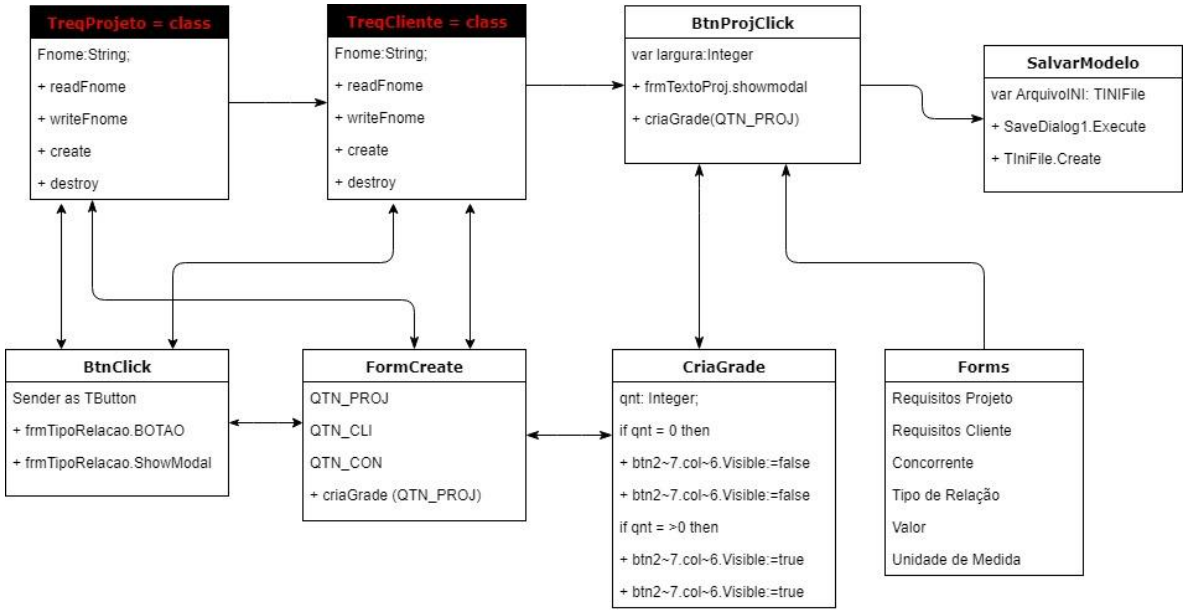
- Disponibilidade para edição em plataforma *desktop* e *mobile*;
- Tipografia sem serifa adequada para telas;
- Interface amigável e que possibilita editar os campos diretamente nas matrizes;
- Botões de tamanho adequado para serem acionados com *mouse* ou com o dedo;
- No caso da versão *mobile*, posicionamento dos *forms* na parte superior da tela, evitando com que o teclado virtual atrapalhe sua leitura e preenchimento ao surgir na porção inferior da tela;
- Capacidade de compartilhamento do aplicativo e de arquivos QFD em nuvem.

4.3 Processo de programação

O aplicativo foi desenvolvido utilizando a linguagem de programação Object Pascal, que é multifacetada, combinando o poder da programação orientada ao objeto, suporte para programação genérica e construções dinâmicas como atributos, mas sem remover o suporte para o estilo tradicional de métodos de programação (CANTÙ, 2016). Portanto, foram combinadas as estratégias de programação, alternando e mesclando a POO à programação estrutural, variando-as de acordo com a necessidade e facilidade de implementação de cada uma.

A Figura 3 apresenta o diagrama de classes elaborado para a criação do *software*. Utilizando os princípios básicos da POO, foram isoladas duas únicas classes do programa: Requisitos de Projeto e Requisitos do Cliente. Cada classe possui botões associados para a criação das suas respectivas células (BtnClick e BtnProjClick), que através do FormCreate e CriaGrade forma a matriz de correlações no formato adequado. Juntamente com a criação do telhado da casa da qualidade, torna-se possível editar os concorrentes e todos os demais dados do QFD, através dos *Forms*. Por fim, todas as informações são salvas através do SalvarModelo.

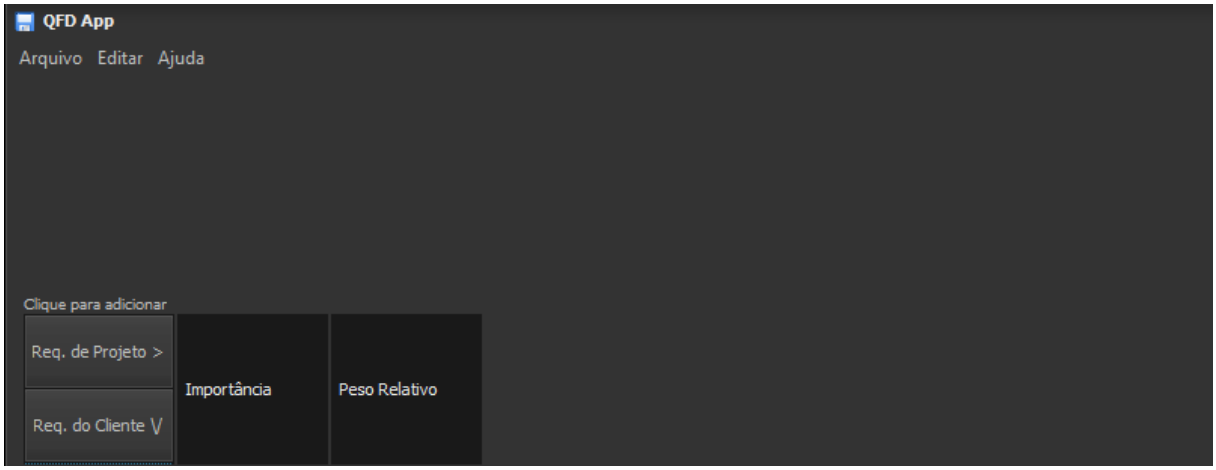
Figura 3 - Diagrama de classes do *software*



Fonte: autores

A tela inicial do aplicativo é minimalista, apresentando somente os botões para a inclusão dos Requisitos de Projeto e Requisitos do Cliente (Figura 4). Essa estratégia permitiu uma tela inicial limpa, o que facilita o uso, especialmente em dispositivos móveis. Cada vez que esses botões são acionados, é criada uma linha ou coluna, sendo o texto do *form* enviado à *Grid Principal*, computando as linhas e colunas atuais e adicionando a próxima. Dessa forma, são lançados os dados do texto digitado e também os dados relativos à quantidade de colunas ou linhas, com tamanhos pré-especificados para que o telhado da casa da qualidade se forme nas posições adequadas. As variáveis globais QTN_PROJ e QTN_CLI são utilizadas para definir a quantidade de linhas e colunas, visando a criação do próximo requisito após o último existente.

Figura 4 - Leiaute da tela inicial



Fonte: autores

O menu superior é o Componente *MyMenu,* e nele são carregados os ícones salvos na pasta *imgs*. Foram incluídas no menu as funções de **Arquivo**>Salvar, Abrir e Sair, a função **Editar**> Limpar Tela e **Ajuda**.

Foram definidos três *grids* como base para o programa: Principal, Concorrentes e Auxiliar. No principal estão agregados os botões para criação dos Requisitos de Projeto e do Cliente e suas respectivas células, formadas por linhas e colunas. Quando se cria o Requisito do Cliente, ao seu lado pode ser cadastrada sua importância, que será valorada em porcentagem na célula seguinte. As *grids* Concorrentes e Auxiliar ficam inicialmente ocultas, aparecendo somente após a inclusão do primeiro Requisito de Projeto. Em seguida, cada célula pode ser editada e, para isso, há um evento global, *ONSELECTCELL* abrangendo todas as células. Nesse evento é tratado qual *form* vai ser aberto, de acordo com a função, através da contagem de linhas e colunas.

Ainda na Grid Principal estão alocados todos os botões que formam o telhado da casa da qualidade. Os botões possuem um *offset* de 50% em relação às colunas de Requisitos de Projeto, sendo exibidos na medida em que esses vão sendo criados. Os mesmos estão limitados a seis requisitos, em virtude do leiaute do programa levar em conta tamanhos de telas de *notebooks* e *tablets*, para evitar constantes rolagens para o preenchimento do QFD. A partir da quantidade de Requisitos de Projeto, se determina a quantidade de botões a serem exibidos, através do comando *Visible:=true.*

O Grid Concorrentes analisa os *benchmarkings* competitivos em relação ao produto próprio, caso ele já exista. Já o Grid Auxiliar é formado abaixo da matriz de relações, em que estão as células para quantificação dos Requisitos do Produto, que estabelecem a dificuldade técnica para atendimento dos Requisitos de Projeto, os valores meta e suas respectivas unidades de medida.

Com o programa em funcionamento, a próxima etapa determinou o tipo de arquivo para o salvamento do QFD, para recuperações e edições futuras. A primeira opção foi salvar o arquivo em formato TXT, porém, a importação de dados se mostrou complexa. Outra opção viável foi armazenar em um arquivo de conteúdo de configuração, em formato INI. Esse formato possui uma estrutura padrão constituída de seções, propriedades e valores (DELPHI – TRABALHANDO COM ARQUIVOS INI, 2017):

- As **seções** representam agrupamentos de valores, ou seja, o assunto no qual se destina os valores armazenados.
- As **propriedades** são os nomes das configurações que se deseja armazenar (Requisitos de Projeto, Importância, etc).
- Os **valores**, como o próprio nome diz, armazenam os dados das configurações (Forte, Fraca, 50 °C, Negativa Forte, etc).

Todos os dados preenchidos no QFD são salvos no cabeçalho do arquivo, que possui valores específicos, sendo mais fácil e rápida a leitura. Nos testes realizados, mesmos em computadores com *hardware* limitado, os arquivos abriram de imediato, sem demora no carregamento.

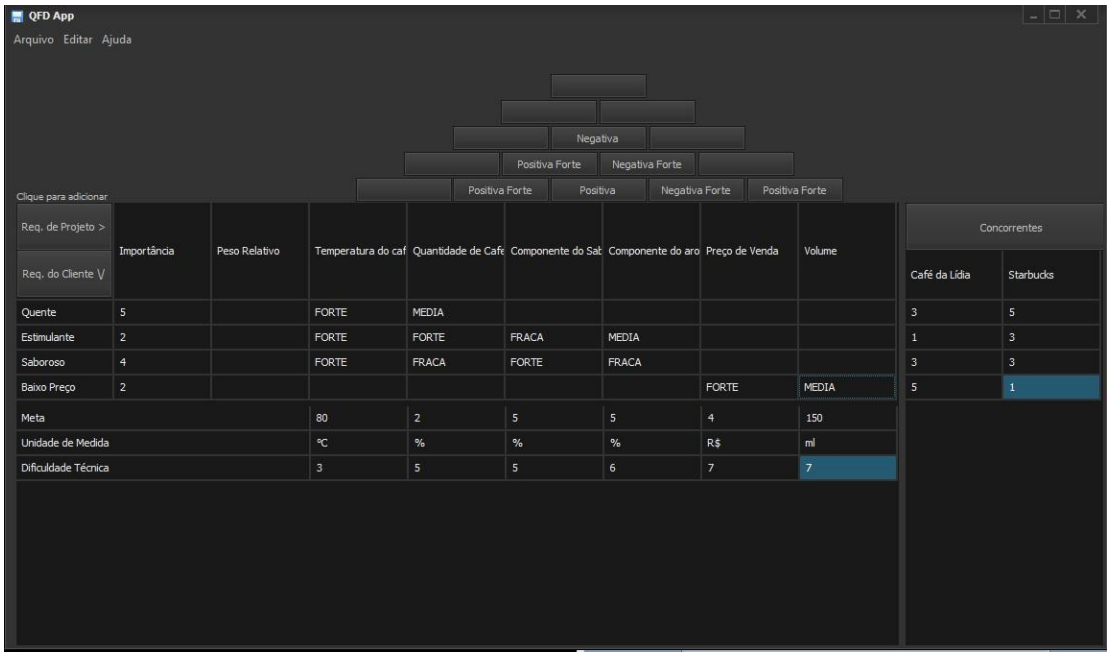
4.4 Protótipos e testes

A partir do protótipo desenvolvido foram realizados testes iniciais de usabilidade envolvendo tarefas fundamentais de aplicativos QFD, bem como tarefas essenciais ao escopo deste trabalho, tais como: preencher os campos do QFD para o desenvolvimento de um novo produto; abrir e utilizar o aplicativo em plataformas *desktop* e *mobile*; abrir arquivos existentes e salvar arquivos de QFD.

Nesse sentido, observou-se a capacidade do *app* em atender aos princípios universais de usabilidade – eficácia, eficiência e satisfação (ABNT, 2002). Além disso, foram estabelecidos comparativos entre o protótipo e os aplicativos similares analisados previamente.

O protótipo foi testado em dispositivos que rodam os sistemas operacionais Windows 8 e 10, tanto em modo *desktop* quanto *tablet*, apresentando bons resultados em ambos os ambientes. Para fins de comparação, foi executado o QFD do produto cafezinho (Figura 5), em conformidade com o que foi realizado anteriormente durante a etapa de análise de similares.

Figura 5 - QFD Cafezinho no aplicativo desenvolvido



Fonte: autores

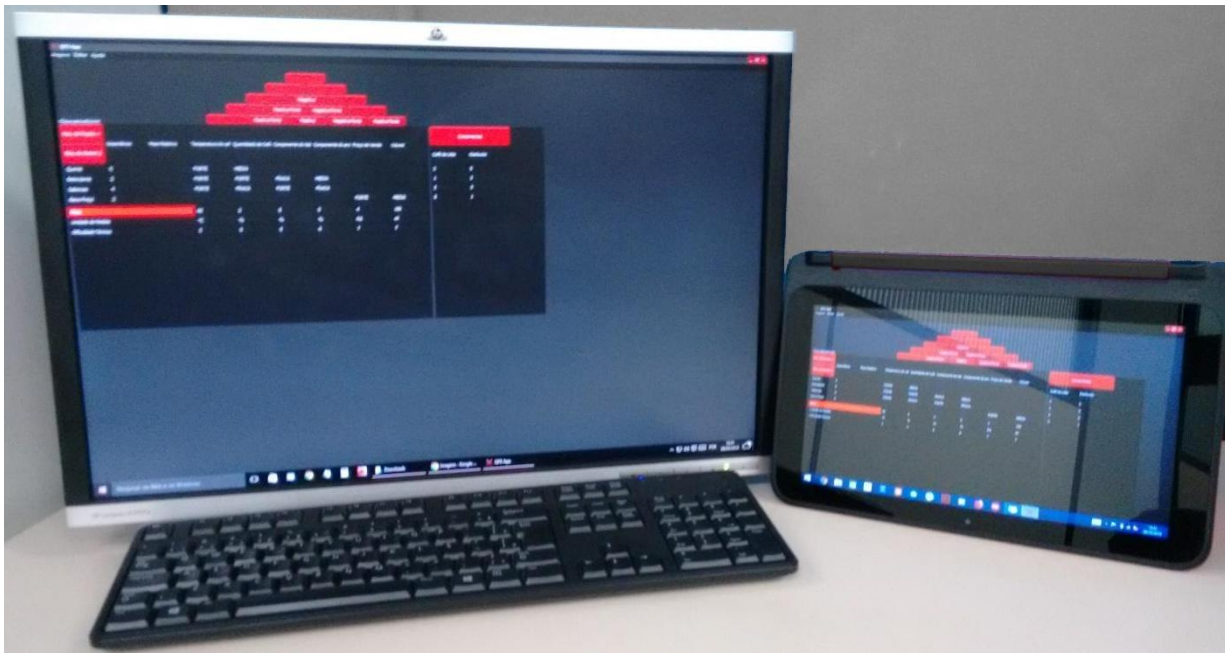
O aplicativo foi projetado tendo em vista sua facilidade de uso, o acesso e a compatibilidade entre diferentes plataformas, assim como a capacidade de compartilhamento. Dessa forma, desenvolveu-se uma interface simples, com cores e elementos contrastantes, e uso de tipografia sem serifa em tamanho legível. O uso das matrizes ocorre de maneira intuitiva: requisitos de cliente e requisitos de usuário são incluídos com cliques em botões de fácil acionamento; os campos das matrizes podem ser editados diretamente com o clique do *mouse* ou toque do dedo; as relações entre os requisitos de cliente e requisitos de projeto são estabelecidas através de palavras (fraca, média, positiva, negativa, etc.), o que agiliza a compreensão.

O usuário tem a flexibilidade de utilizar a ferramenta para atingir objetivos de menor ou maior complexidade, tais como estabelecer e hierarquizar os requisitos de projeto; determinar especificações (metas) de projeto e assim por diante. Dessa forma, também é possível utilizar somente aqueles campos destinados a essas operações dentre as matrizes disponíveis.

Considerando os aspectos de compartilhamento, o aplicativo foi compilado em versão executável (.exe), sem a necessidade de instalação prévia e, portanto, sem a exigência de privilégios de administrador, tendo tamanho de 12 megabytes, e podendo facilmente ser compartilhado via e-mail com colaboradores. Esses recursos possibilitam o salvamento e compartilhamento em nuvem do QFD, o que pode ser feito com o uso de plataformas universais, tais como Google Drive, Microsoft OneDrive e Dropbox.

O protótipo desenvolvido proporcionou um uso prático e sem dificuldades, apoiando tarefas de criação, edição, salvamento e recuperação de arquivos, tanto em plataforma *desktop* como *mobile*. Todas as tarefas básicas estabelecidas no teste de usabilidade foram executadas sem maiores dificuldades. Quando comparado com outros aplicativos analisados, o tempo de criação de um QFD mostrou-se similar. O uso no dispositivo *tablet* mostrou-se superior à outra versão móvel concorrente (Figura 6).

Figura 6 - QFD sendo utilizado em modo *desktop* e *tablet*



Fonte: autores

5 Considerações Finais

Este trabalho foi desenvolvido a partir de uma lacuna identificada no mercado dos aplicativos de QFD e das tendências para a mobilidade e trabalho remoto, bem como da necessidade de otimizar o PDP com o uso de ferramentas para qualificar o processo de projeto (BOMFIM, 1995; LÖBACH, 2001; BOSOMWORTH, 2015; HERZWURM; SCHOCKERT; MELLIS, 1997; ANDRIOLLI, 2018). O modelo de Rozenfeld *et al.* (2006), utilizado como referência nas etapas iniciais do processo de desenvolvimento, posiciona este aplicativo como uma ferramenta específica para o projeto de produtos. Outros modelos e programas analisados servem a propósitos mais amplos, como a gestão de qualidade, gestão de processos produtivos, entre outros. Dessa forma, o aplicativo aqui proposto torna-se diretamente relevante às universidades e MPEs focadas no projeto de produtos.

O processo de desenvolvimento utilizado contribuiu para a criação de um aplicativo de fácil utilização, multiplataforma e de código aberto. A análise do QFD no processo de desenvolvimento de produtos permitiu estabelecer as principais funcionalidades. Ainda, a análise de aplicativos similares possibilitou conhecer melhor os requisitos e restrições do novo aplicativo. Quando comparado com demais, a ausência de custo, a facilidade no uso e o potencial de desenvolvimento futuro através da POO se destacam. *Softwares* pagos costumam ter interfaces similares às planilhas eletrônicas e sem opções de compatibilidade entre *desktop* e *mobile*. Planilhas eletrônicas, apesar de serem familiares aos usuários, são construídas com propósitos variados e, considerando que não foram especificamente programadas para o uso de QFDs, têm restrições na interface e funções desempenhadas.

Decisões projetuais simples como a possibilidade de edição automática nos campos e o uso de palavras em vez de símbolos contribuíram para a usabilidade do aplicativo, verificada em testes iniciais com protótipos. A possibilidade de trabalho em multiplataforma de maneira colaborativa

abre possibilidades para o trabalho em equipes de projeto e adiciona segurança no caso de possíveis perdas de arquivo decorrentes de problemas de *hardware*. Visto que cada vez mais dispositivos móveis são utilizados em empresas e universidades, torna-se possível a utilização dessa ferramenta em qualquer ambiente, não ficando necessariamente restrito a escritórios ou laboratórios. Isso abre ainda mais o potencial do QFD como uma ferramenta de comunicação, como é salientado por Morris (2011), tornando-se possível discutir sobre requisitos de usuário ou de projeto em contextos mais descontraídos que estimulam a criatividade.

No decorrer do desenvolvimento e dos testes, bem como após a experiência com outros métodos de criação de QFDs, percebem-se potenciais desenvolvimentos no aplicativo:

- Melhora na identificação das colunas ou linhas que estão sendo classificadas nas matrizes de relação e correlação, possivelmente utilizando eventos *On Mouse Over* para salientar os dados envolvidos.
- Desdobramento sequencial dos QFDs: Os requisitos do projeto criam o segundo QFD, que por sua vez geram o terceiro e assim sucessivamente. Todos os QFDs devem ficar interconectados e herdam qualquer alteração dos arquivos prioritários, o que os associa aos conceitos de engenharia simultânea e/ou concorrente. Assim, as decisões de projeto são focadas cada vez mais nos detalhes de fabricação do produto (BAXTER, 2010).
- Compatibilidade e/ou associação com outras ferramentas de seleção, como o método Kano e matriz de Pugh, conforme sugerido por Rozenfeld *et al.* (2006).
- Possibilidade de salvar em um formato específico para o aplicativo; exportação para *softwares* de leitura genéricos (PDF), ou de edição, como planilhas eletrônicas.
- Compilar o programa para *tablets* Android e IOS.
- Aprimorar a interface para uso também em celulares.
- Os formulários que indicam as correlações (FRACA, MÉDIA, FORTE ou N/A) poderão ter valores numéricos atribuídos a eles posteriormente (1 a 5), tornando possível fazer a soma em uma linha para valoração da máxima relação de valor nas linhas ou colunas.

Observa-se que a POO auxilia na organização e limpeza do código de programação, facilitando o entendimento e edição do programa, sendo esse conceito utilizado principalmente na criação das classes Requisitos de Projeto e do Cliente. Além disso, o emprego de um único *form* para valoração de todas as células e telhado da casa da qualidade elimina várias linhas de código, tornando o programa mais leve e rápido. Essa configuração permite que, por exemplo, as sugestões citadas acima sejam implementadas de forma mais rápida, levando em consideração, ainda, que certas funções seriam impossíveis de serem implementadas em uma planilha eletrônica.

5 Referências

ABNT. NBR 9241-11. **Requisitos ergonômicos para trabalho de escritório com computadores**: Parte 11 - Orientação sobre usabilidade. ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. Rio de Janeiro: sn, p. 21, 2002.

ANDRIOLLI, A. **Trabalho Remoto:** como usar tecnologia para garantir produtividade, 2018. (http://computerworld.com.br/trabalho-remoto-como-usar-tecnologia-para-garantir-

produtividade)

AKAO, Y. **Quality Function Deployment:** Integrating Customers Requirements into Product Design. Cambridge, Massachusetts: Productivity Press, 1990.

BACK, N. *et. al.* **Projeto integrado de produtos:** planejamento, concepção e modelagem. Barueri, SP: Manole, 2008.

BAXTER, M. **Projeto de produto:** guia prático para o design de novos produtos. 2. ed. São Paulo: Edgard Blücher, 2010.

BERDONOSOVA, V. ZHIVOTOVAB, A. SYCHEVA, T. **TRIZ evolution of the Object-Oriented Programming Languages.** Procedia Engineering 131 (2015), 333 – 342.

BOMFIM, G. **Metodologia para Desenvolvimento de Projeto**. João Pessoa: Editora da UFPB, 1995.

BOSOMWORTH, D. **Mobile Marketing Statistics.** 2015. (http://plenuminvest.dk/docs/mobile-marketing-statistics-2015.docx)

BRASIL. Lei Nº 13.467, de 13 de Julho de 2017, 2018. (http://legislacao.planalto.gov.br/legisla/legislacao.nsf/Viw_Identificacao/lei%2013.467-2017?OpenDocument)

BUDD, T. **An Introduction to Object Oriented Programming.** Reading, Massachusetts: Addison-Wesley, 1997.

CANTÙ, M. **Object Pascal Handbook.** Piacenza, Italy: 2016.

CHAN, L.; WU, M. **Quality function deployment:** A literature review. European Journal of Operational Research, n.º 143, p. 463–497, jan 2002.

Delphi – Trabalhando com Arquivos INI, 2017. (http://www.andrecelestino.com/delphi-trabalhando-com-arquivos-ini)

HERZWURM, G., SCHOCKERT, S., MELLIS, W., 1997. **Customer oriented evaluation of QFD software tools**. In: Transactions of the Third International Symposium on Quality Function Deployment, October 1–2, Linkoping, Sweden, € vol. 1.

House of Quality App, 2017. (http://dbis.rwth-aachen.de/apps/HouseOfQuality)

LÉVY, Pierre. **Cibercultura**. 3 ed. São Paulo: Editora 34, 2010.

LÖBACH, B. **Design industrial:** bases para a configuração dos produtos industriais. São Paulo: Edgard Blucher, 2001.

MORRIS, Richard. **Fundamentos de design de produto**. Porto Alegre: Bookman, 2011.

NETTO, A. **Novas tecnologias & universidades**: da didática tradicionalista à inteligência artificial: desafios e armadilhas. Petrópolis: Vozes, 2005.

Passo a passo para construção da casa da qualidade, 2017. (http://www.blogdaqualidade.com.br/passo-a-passo-para-a-construcao-da-casa-da-qualidade-qfd)

QFD Capture, 2018. (http://www.qfdcapture.com)

QFD House Of Quality, 2018. (https://itunes.apple.com/br/app/qfd-house-of-

quality/id1016279817?mt=8)

PRASAD, B. **Reviews of QFD and Related Deployment Techniques.** Journal of Manufacturing Systems. Vol. 17/No. 3, 1998.

ROZENFELD, H. *et al.* **Gestão de desenvolvimento de produtos:** uma referência para a melhoria do processo. São Paulo: Saraiva, 2006.

SIVALOGANATHAN, S; EVBUOMWAN, N. **Quality Function Deployment - The Technique:** State of the Art and Future Directions. Concurrent Engineering: Research & Applications, Vol. 5/No. 2 (1997), 171 - 182.