



APLICAÇÃO DA TÉCNICA KANO PARA PRIORIZAÇÃO DE REQUISITOS EM UM PROJETO ÁGIL

Trabalho de Conclusão de Curso

Engenharia da Computação

Luiza Freire Paiva Alves Lira

Orientador: Prof.^a Maria Lencastre Pinheiro de Menezes Cruz



**UNIVERSIDADE
DE PERNAMBUCO**

**Universidade de Pernambuco
Escola Politécnica de Pernambuco
Graduação em Engenharia de Computação**

LUIZA FREIRE PAIVA ALVES LIRA

**APLICAÇÃO DA TÉCNICA KANO PARA
PRIORIZAÇÃO DE REQUISITOS EM UM
PROJETO ÁGIL**

Monografia apresentada como requisito parcial para obtenção do diploma de Bacharel em Engenharia de Computação pela Escola Politécnica de Pernambuco – Universidade de Pernambuco.

Recife, dezembro de 2017.

A Deus por me sustentar nessa caminhada, ao meu marido Renato Lira por todo amor e compreensão, ao meu filho Leonardo que mesmo no ventre me impulsiona a seguir em frente, aos meus pais e a toda minha família que com muito carinho e apoio, não mediram esforços para que eu chegasse até aqui, aos queridos professores e mestres que tanto se empenharam na construção da minha carreira profissional, aos amigos que a universidade me proporcionou ter, os guardarei no coração. A todos meu muito obrigada!

Resumo

A competição acirrada tem levado as empresas de *software* a adotar metodologias de desenvolvimento ágeis, com objetivo de garantir maior competitividade e crescimento da satisfação dos clientes. Em contrapartida, a priorização de requisitos é considerada uma atividade crítica do desenvolvimento de software, que envolve uma análise criteriosa da importância de cada requisito por parte dos *stakeholders* e a seleção dos requisitos que serão implementados na próxima versão do sistema. Uma decisão errada sobre quais requisitos priorizar pode afetar a qualidade geral do sistema, e como consequência sua aceitação pelos clientes. No desenvolvimento ágil é mais enfatizada a indicação do cliente para realizar a priorização dos requisitos. Porém, nela alguns problemas podem ocorrer, como o cliente acreditar que um requisito possui mais importância do que ele realmente possui, assim como ele pode não dar a devida importância a um requisito que ele desejava muito, mas que só sentirá necessidade quando o sistema estiver concluído e com essa funcionalidade faltando. Metodologias ágeis como o *Scrum* e *Extreme Programming* são capazes de oferecer orientações bastante básicas sobre como conduzir essa priorização. No entanto, dentre as principais limitações do processo de priorização de requisitos para projetos em ambientes ágeis estão a dificuldade de comparar a real importância dada aos requisitos como também a falta de análise caso determinados requisitos não sejam selecionados. O objetivo central deste trabalho é propor a estruturação do processo de priorização de requisitos para projetos de software ágeis, tomando como base a técnica Kano, originalmente proposta na área da administração; os diagramas oferecidos pela técnica objetivam priorizar os itens através de uma pergunta funcional e uma pergunta disfuncional para cada requisito. O processo proposto nesta monografia visa auxiliar a priorização de requisitos melhorando a forma de visualizar os resultados da priorização através de gráficos e quadros mais ilustrativos, facilitando a tomada de decisão por parte dos *stakeholders* do projeto. Com objetivo de avaliar a adequação da proposta, foi realizado um estudo de caso em uma empresa de software em Recife; os resultados encontrados sugerem que a abordagem proposta facilita a priorização de requisitos.

Abstract

The fierce competition has led software companies to adopt agile development methodologies, with the goal of ensuring greater competitiveness and customer satisfaction growth. On the other hand, the prioritization of requirements is considered a critical activity of software development, which involves a careful analysis of the importance of each requirement by the stakeholders and the selection of requirements that will be implemented in the next version of the system. A wrong decision about which requirements to prioritize can affect the overall quality of the system, and as a consequence its acceptance by customers. In agile development, more emphasis is placed on the indication of the client to perform the prioritization of the requirements. But in it some problems may occur, as the client believes that a requirement has more importance than it actually has, just as it may not give due importance to a requirement that it has longed for, but which will only feel need when the system is completed and with this functionality missing. Agile methodologies with Scrum and Extreme Programming are able to offer fairly basic guidance on how to drive this prioritization. However, among the main limitations of the requirements prioritization process for agile environments are the difficulty of comparing the real importance given to the requirements as well as the lack of analysis if certain requirements are not selected. The main objective of this work is to propose the structuring of the requirements prioritization process for agile software projects, based on the Kano technique, originally proposed in the administration area; the diagrams offered by the technique aim to prioritize the items through a functional and dysfunctional question for each requirement. The process proposed in this monograph aims to support the prioritization of requirements by improving the way of visualizing the results of prioritization through more illustrative charts and tables, facilitating the decision-making of project stakeholders. In order to evaluate the adequacy of the proposal, a case study was carried out at a software company in Recife; the results suggest that the proposed approach facilitates the prioritization of requirements.

Sumário

1. Introdução	12
1.1 Motivação	12
1.2 Descrição do Problema	13
1.3 Metodologia adotada	14
1.4 Organização da Monografia	15
2. Metodologias Ágeis	16
2.1 Valores Ágeis	17
2.1.1 Comunicação	18
2.1.2 Participação do Cliente	18
2.3 Scrum	19
2.3.1 Papéis do Scrum	19
2.3.2 Fluxo do Scrum	20
2.4 Considerações Finais	22
2. Engenharia de Requisitos	23
3.1 Conceituação	23
3.2 Etapas da Engenharia de Requisitos	25
3.2.1 Elicitação de Requisitos	26
3.2.2 Análise e Negociação dos Requisitos	27
Priorização de requisitos	28

3.2.3	Documentação dos Requisitos	29
3.2.4	Validação dos Requisitos	30
3.2.5	Gerenciamento dos Requisitos	31
3.3	Técnicas de Priorização	33
3.3.1	Visão Geral das Técnicas de Priorização de Requisitos	33
3.3.2	Priorização utilizando Kano	34
3.3.3	Como Utilizar Kano	37
3.4	Considerações Finais	39
4.	Priorização de Requisitos em Ambientes Ágeis	40
4.1	Novos elementos visuais para Kano	40
4.2	Etapas da Priorização	42
4.3	Estudo de Caso	46
4.3.1	Aplicação da Técnica Kano	46
4.4	Considerações Finais	55
5.	Conclusão e trabalhos futuros	57
	Bibliografia	58
	Apêndice	61

Índice de Figuras e Tabelas

Figura 1.	Custo relativo para corrigir um defeito.....	24
Figura 2.	Atividades no modelo espiral do processo de ER	26
Figura 3.	Diagrama de Kano sobre satisfação do cliente.	36
Figura 4.	Resultado das combinações às respostas funcionais e disfuncionais..	38
Figura 5.	Gráfico em forma de colunas da Tabela 2.....	40
Figura 6.	Visualização do gráfico de pizza do Requisito 1.....	41
Figura 7.	Visualização do gráfico pizza do Requisito 2.....	42
Figura 8.	Visualização do gráfico pizza do Requisito 3.....	42
Figura 9.	Fases da proposta de priorização de requisitos	43
Figura 10.	Opções de respostas ao questionário	43
Figura 11.	Utilização do gráfico para análise.....	44
Figura 12.	Questionário aplicado utilizando Excel.....	46
Figura 12.	Exemplo de questionário aplicado aos usuários.....	47
Figura 13.	Visualização dos dados obtidos no gráfico por colunas.	49
Figura 14.	Gráfico em pizza (Requisito 1)	50
Figura 15.	Grafico em pizza (Requisito 2)	50
Figura 16.	Gráfico em pizza (Requisito 3)	51
Figura 17.	Gráfico em pizza (Requisito 4)	52

Figura 18.	Gráfico em pizza (Requisito 5)	52
Figura 19.	Gráfico em pizza (Requisito 6)	53
Figura 22.	Gráfico em pizza (Requisito 7)	54
Figura 20.	Gráfico em pizza (Requisito 8)	55

Índice de Tabelas

Tabela 1. Algumas das principais técnicas de priorização de requisitos.	33
Tabela 2. Simulação dos resultados obtidos.....	38
Tabela 3. Resposta obtida dos usuários	48
Tabela 4. Somatório das respostas dos usuários classificada por requisito.	48
Tabela 5. Percentual de respostas obtidas por requisito.	48
Tabela 6. Sugestão de priorização para próxima entrega.....	55

Lista de Símbolos e Siglas

PO – Product Owner

ER – Engenharia de Requisitos

FDD – Feature Driven Development

XP – Extreme Programming

AHP - Analytic Hierarchy Process

1. Introdução

Este capítulo apresenta as principais motivações para a realização desta monografia, buscando contextualizar o objetivo principal da pesquisa e descrever a motivação para realização da pesquisa, metodologia utilizado, o objetivo que pretende-se alcançar, bem como, a estruturação do trabalho.

1.1 Motivação

O sucesso dos sistemas depende basicamente de critérios simples, como: atender o prazo, escopo e custo estabelecidos na fase inicial do projeto acordados com o cliente. Segundo (Lamsweerde, 2002) satisfazer os requisitos do cliente é um dos fatores mais importantes para o sucesso do desenvolvimento de sistemas; para que esse objetivo seja atendido é necessário que o projeto seja entregue com os requisitos que tragam maior satisfação para o cliente, aumentando assim o valor de mercado do sistema. Por isso, podemos dizer que a priorização de requisitos é uma das atividades mais críticas de todo processo de desenvolvimento de software, que envolve análise minuciosa da importância de cada requisito elicitado por parte dos *stakeholders*; qualquer decisão errada poderá afetar a qualidade global do sistema e sua aceitação perante os clientes ou usuários.

Em contrapartida, a necessidade das empresas de software de desenvolver os projetos cada vez mais rapidamente e de forma eficiente, fez com que elas adotassem Metodologias Ágeis para garantir uma vantagem competitiva e conseqüentemente, aumentar a satisfação dos clientes. Por isso, nos últimos anos diversas metodologias ágeis surgiram na literatura, tais como: *Scrum*, *Feature Driven Development (FDD)*, *Extreme Programming (XP)*.

Podemos perceber também uma grande dificuldade em se escolher a ordem da implementação dos requisitos, sejam na metodologia adotada ágil ou na tradicional. Desta forma, a priorização de requisitos é um problema real enfrentado

por empresas de software, além de ser fortemente discutido na literatura a sua importância, como descrito por (Cockburn, 2002).

1.2 Descrição do Problema

Diante da realidade moderna e frenética em que estamos inseridos, é difícil prever como um sistema computacional se comportará com o passar do tempo. As necessidades do usuário final estão em constante mudança, as tecnologias presentes no mercado se tornam obsoletas em um curto intervalo de tempo. Diante disso, definir todos os requisitos de uma aplicação se torna praticamente inviável nos momentos iniciais do projeto.

Aliado a essa constante mudança, existe também um grande aumento da complexidade e tamanho das aplicações atuais, onde a priorização de requisitos de software passa a ser uma atividade fundamental no desenvolvimento de sistemas, e na viabilização do sucesso do produto. Porém, geralmente a priorização de requisitos é uma atividade realizada muito tarde no processo de desenvolvimento de software; além disso, nem todos os clientes têm facilidade em priorizar requisitos, pois na maioria das vezes consideram todos os requisitos como sendo de alta prioridade.

Conforme apresentado por (Junior, 2015), numa pesquisa realizada na cidade de Recife em 2015 em 39 empresas, existe uma grande resistência em se aplicar técnicas de priorização de requisitos; a maioria das instituições faz a priorização de uma forma intuitiva, por achar que a aplicação das técnicas de priorização é uma perda de tempo, cara e inútil. Estas empresas não se apercebem do impacto da não realização de uma priorização de requisitos mais precisa, considerando a avaliação de critérios que ajudam na melhoria da eficiência do desenvolvimento e na satisfação do cliente. De acordo com (Junior, 2015), pode-se observar que essa realidade se estende a empresas nacionais e internacionais.

Por outro lado, segundo (Pressman, 2011), “É preciso ser ágil suficiente para dar uma resposta ao ambiente fluido de negócios”. De tal forma, as metodologias

Ágeis predominam nos ambientes de desenvolvimento de software da atualidade. Em ambientes de desenvolvimento ágil, versões da aplicação devem ser entregues frequentemente e de forma incremental e, a cada nova versão, deve-se selecionar os requisitos de maior prioridade a serem implementados. No entanto, a maioria dos projetos de desenvolvimento de software pode não capturar de forma adequada a necessidade do cliente, para indicação de qual requisito deveria ser implementado primeiro, o que poderá gerar alguns problemas tais como inclusão de vários requisitos com prioridade máximo, de forma tal, que não seja possível determinar a real importância de cada requisito. Segundo (Asfora, 2009), no caso da metodologia ágil Scrum, os clientes classificam a importância dos requisitos numa escala de 0 a 1000, e com frequência se tem vários requisitos classificados como 1000.

1.3 Metodologia adotada

Este estudo tem por propor uma aplicação prática da técnica Kano, como novos elementos visuais e gráficos, adaptado para um Projeto Ágil, com intuito de propor uma solução para deficiência visual da técnica Kano (Kohn, 2005), por isso é de natureza aplicada. Neste caso, são usados conhecimentos existentes sobre Metodologias Ágeis e priorização de requisitos, a fim de propor critérios e formas para auxiliar a utilização conjunta da priorização durante a aplicação de Metodologias Ágeis.

No estudo focamos no uso da técnica Kano, no intuito de tornar mais eficaz a definição da prioridade das atividades nos processos relativos ao desenvolvimento utilizando Scrum. (Schwaber, 2004). Como uma forma de enriquecer o trabalho, foi também investigada a existência de trabalhos que já façam essa integração. A restrição do escopo a uma técnica de priorização e a uma metodologia ágil visa viabilizar a proposta no tempo previsto;

Nesta pesquisa, foi adotada uma abordagem qualitativa. Qualitativa porque visa aprofundar o conhecimento do problema, buscando entendê-lo através da definição de critérios e formas de aplicação e por um processo de inferência

dedutiva, o qual testa a predição da ocorrência de fenômenos abrangidos pela referida hipótese (Marconi e Lakatos, 2003).

1.4 Organização da Monografia

Esta monografia encontra-se organizada em cinco capítulos. O presente capítulo apresenta uma introdução sobre as motivações, descrição do problema e as contribuições deste estudo.

No Capítulo 2, são introduzidas as Metodologias Ágeis e seus princípios e valores. Além disso, é apresentada a Metodologia *Scrum*.

No capítulo 3, são introduzidos os conceitos e técnicas utilizadas na Engenharia de Requisitos; além disso, é apresentada a técnica de priorização de requisitos Kano.

No capítulo 4, é apresentada a proposta para utilização da técnica Kano aplicada ao *Scrum*.

No capítulo 5, são apresentadas as conclusões e trabalhos futuros.

Por fim, no Apendice A são apresentados os resultados consolidados referentes ao estudo de caso realizado.

2. Metodologias Ágeis

Este capítulo apresenta a Metodologia Ágil e descreve as técnicas e abordagens utilizadas para gerenciar os projetos de desenvolvimento de software em ambientes ágeis.

As abordagens para ambientes ágeis foram formalmente apresentadas para a comunidade acadêmica através do manifesto ágil em 2000 [Manifesto Ágil]. O manifesto ágil estabelece um conjunto de valores que são adotados em projetos ágeis, são estes:

- **Software funcionando**, ao invés de **documentação abrangente**;
- **Indivíduos e interações** ao invés de **processos e ferramentas**;
- **Colaboração com o cliente** ao invés de **negociação de contratos**;
- **Responder a mudanças** ao invés de **seguir um plano**;

Atualmente, alguns dos principais processos ágeis conhecidos são: *Scrum* (Schwaber, 2004), *Feature Driven Development (FDD)* (Palmer, 2002), *Extreme Programming* (Johnson, 2002). Neste estudo, focaremos no método *Scrum*, por ser um dos métodos mais utilizado nas empresas de tecnologia da informação.

O diferencial entre um processo de desenvolvimento tradicional e o ágil, é o que podemos chamar de *barely sufficient*, ou seja, o mínimo necessário. Enquanto as abordagens tradicionais, como Rational Unified Process (RUP, 2002), procuram definir um guia de boas práticas, os processos ágeis definem poucas práticas. Isto torna os ambientes ágeis mais simples de serem adotados. A ênfase encontra-se em trazer elementos novos para o projeto somente quando os mesmos realmente se mostram necessários. (Highsmith, 2002).

Na próxima sessão deste trabalho são apresentados os principais valores ágeis e é descrita a abordagem ágil *Scrum*.

2.1 Valores Ágeis

Um dos mais importantes princípios ágeis é obter *feedback*, interpretá-lo e colocar o que foi aprendido de volta no processo de desenvolvimento no projeto o mais rápido possível. A compreensão das necessidades dos usuários é um processo de aprendizado contínuo, em que os desenvolvedores vão aprender sobre o problema do negócio e os usuários; em contrapartida eles devem estar cientes das dificuldades e limitações técnicas. Segundo (Weinberg, 1971) para atingir essa compreensão dos usuários, existe um princípio psicológico muito conhecido que diz que para maximizar a taxa de aprendizado, o indivíduo precisa receber *feedbacks* sobre quão bem ou mal ele está indo.

Através desse princípio de *feedback* contínuo, metodologias ágeis como *Scrum* se utilizam do conceito de *Sprints*, ou seja, iterações que possuem tempo curto. Esse processo, envolve a priorização de poucas funcionalidades a serem implementadas de cada vez e simplificá-las na medida do possível. O objetivo é apresentar a funcionalidade ao usuário o mais rápido possível, de modo que ele possa, o mais cedo possível detectar possíveis falhas, enquanto ainda é barato corrigi-las. A principal razão para a utilização de estratégias incrementais e iterativas é permitir que os inevitáveis erros sejam descobertos relativamente cedo e reparados metodicamente (Cockburn, 2002).

Com esses ciclos de *feedbacks* curtos o *Scrum* procura assegurar que uma pouca quantidade de trabalho seja efetuada e concluída de uma só vez. Caso tudo esteja correto, o time de desenvolvimento segue com o projeto. A utilização de pacotes reduzidos de trabalho pode assegurar que eventuais falhas sejam corrigidas com maior rapidez, simplesmente porque o escopo do trabalho é reduzido, o que significa que existe uma menor possibilidade das coisas darem errado.

2.1.1 Comunicação

Frequentemente, falhas no processo de comunicação podem causar equívocos e desentendimentos da compreensão correta de algum aspecto do projeto. Por exemplo, a especificação do requisito, que envolve um processo de comunicação e interpretação, explica porque é uma das maiores dificuldades do processo de desenvolvimento de *software*. Transportar conhecimento de um contexto para outro, não envolve apenas conhecimentos técnicos, mas também questões de significado e intenções conceituais e subjetivas.

Segundo (Teles, 2004), a forma de repassar uma ideia exerce uma grande influência na compreensão da mesma. Os projetos ágeis apresentam a comunicação como um grande problema que deve ser muito bem resolvido através da colaboração e envolvimento direto dos usuários (ao menos um representante dos mesmos) como parte integrante da equipe de desenvolvimento. Na prática, isso significa dizer que o usuário deve estar presente no mesmo local onde os desenvolvedores trabalham, permitindo que eles tenham acesso rápido e direto a um especialista no domínio do negócio. Ainda de acordo com (Teles, 2004), isso ajuda a acelerar o fluxo de informações e possibilita que a comunicação seja predominantemente feita através de diálogos presenciais.

A rapidez na comunicação também é um aspecto importante para o sucesso dos projetos de software; a velocidade do progresso do desenvolvimento do software está ligada ao tempo que se leva para transmitir informações de uma pessoa à outra. Outro fator, segundo (Cockburn, 2002), que influencia na qualidade da comunicação é a quantidade de pessoas envolvidas. Na proporção que a equipe aumenta, torna-se mais difícil para as pessoas saberem o que os outros estão fazendo e não sobrepor ou interferir no trabalho do outro. Por isso que projetos ágeis buscam trabalhar com um número reduzido de pessoas em cada equipe.

2.1.2 Participação do Cliente

Para que a conclusão de um projeto de software seja realizada com sucesso, faz-se necessário a participação ativa de todas as partes envolvidas. Tanto o cliente,

quanto o time de desenvolvimento têm um papel fundamental neste contexto. O cliente na definição do que ele deseja, suas necessidades e expectativas; isto ocorre não só nessa fase, como também o acompanhamento do projeto. Porém, nada disso vale se os desenvolvedores não possuírem capacidade técnica para desenvolver conforme ele deseja, o projeto estará fadado ao fracasso. De forma semelhante, se a equipe tiver desenvolvedores excepcionais focados no sucesso do projeto e que fazem sempre o que o cliente deseja, e o cliente não estiver envolvido ou não se preocupar em esclarecer dúvidas de negócio, a equipe corre um grande risco de elaborar um software que nunca será utilizado, isto é, que não se encaixa na necessidade do negócio. Isto significa dizer, que o time de desenvolvimento e o cliente são interdependentes e que o sucesso será alcançado caso haja cooperação entre as duas partes.

2.3 Scrum

A metodologia Scrum foi inicialmente proposta por Ken Schwaber em 1996, como um método ágil que prevê as constantes mudanças durante o processo de desenvolvimento de software. O termo *Scrum* é originado do esporte Rugby, o “Scrum” ocorre quando os jogadores de cada time colaboram entre si numa tentativa de avançar juntos no campo do adversário (Highsmith, 2002).

O Scrum, diferentemente de outras metodologias, se destaca por enfatizar a área de gerenciamento de projetos. Descrevendo um conjunto de práticas, papéis que são bem definidos e adaptáveis, com iterações e entregas incrementais. Esse método permite um acompanhamento melhor do que está acontecendo durante o projeto, facilitando os ajustes dele e possibilitando alcançar os objetivos do projeto de forma ágil.

2.3.1 Papéis do Scrum

O Scrum é dividido em três papéis: ScrumMaster (gerente de projeto), Product Owner (cliente) e o time de desenvolvedores, segundo Schwaber eles possuem as seguintes responsabilidades:

ScrumMaster – responsável por treinar todos os envolvidos no projeto sobre o Scrum e alinhar esse conhecimento com a cultura da empresa, de forma a garantir que todas as práticas e regras descritas pelo Scrum estão sendo respeitadas. Possui um papel equivalente ao de gerente de projetos.

Product Owner (PO) – responsável pela representação dos interesses dos usuários do sistema final e por saber o retorno do investimento (ROI) de cada requisito do sistema. Ele é um dos grandes envolvidos na priorização de requisitos, é PO que prioriza o *Product Backlog* (lista de requisitos) a cada *Sprint* (iteração), garantindo assim que as funcionalidades com maior valor agregado sejam concluídas prioritariamente.

Time – Equipe responsável pelo desenvolvimento das funcionalidades, times auto-gerenciados, auto-organizados e multifuncionais. Transformam os itens do *backlog* em incrementos de funcionalidades dentro das iterações, gerenciando paralelamente o seu próprio trabalho. São coletivamente responsáveis pelo sucesso de cada iteração do projeto, no geral.

2.3.2 Fluxo do Scrum

O início de um projeto Scrum vem a partir de uma visão global do sistema a ser desenvolvido. Esta mesma visão vai se tornando mais clara à medida que o projeto é realizado. O *Product Owner* é responsável por produzir esse documento de forma a maximizar o retorno do investimento. O PO elabora o plano do que será realizado no *Product Backlog (PB)*, que é a lista de todos os requisitos funcionais e não funcionais. O PB é priorizada pelos itens mais desejados, os que geram mais valores são divididos nas entregas propostas.

O trabalho é feito em *Sprints* ou iterações, cada *Sprint* é uma iteração de 30 dias consecutivos de acordo com [Schawaber 2004], mas existe uma grande flexibilidade em relação a essa duração, depende muito da característica do time e do esforço para implementar o produto. Cada *Sprint* para ser iniciado é realizado um *Sprint Planning Meeting*, onde o PO e o time de desenvolvimento se reúnem para planejar sobre o que será feito na próxima *Sprint*, sendo selecionadas as histórias de

usuários com maior prioridade. A *Sprint Planning* não pode ser mais longa do que 8h. Esse controle do tempo, nas metodologias ágeis, é fundamental para o auto-gerenciamiento e principalmente para manutenção do foco em todos os momentos. A *Sprint Planning* é dividida em duas partes. Na primeira parte, podendo durar até 8 horas de duração, para planejamento de uma *Sprint* de um mês o PO apresenta os itens de maior prioridade ao time. O time questiona o conteúdo, as intenções e o significado de cada item do *Product Backlog* para garantir o entendimento deles. Nessas horas iniciais é definido o que vai estar na *Sprint*, logo após vem à definição do time de *Story Point* de cada item do *backlog*. Na segunda parte, o time detalha as tarefas que duram até 8h. Todos os dias o time se reúne em *Daily Scrum* que duram no máximo 15 minutos. Nessas reuniões são respondidas as seguintes perguntas:

- O que eu fiz para o projeto, desde última reunião?
- O que irei fazer até a próxima reunião?
- Quais os impedimentos para conseguir finalizar meus objetivos da *Sprint* e das minhas tarefas?

Normalmente, essa reunião é realizada no modelo *Stand up meeting* o que facilita o cumprimento do tempo, por todos estarem de pé e a comunicação é um valor abordado nesse momento. Proporcionando uma troca de conhecimento entre equipe resultando numa sincronização do trabalho do time.

Ao final da *Sprint*, o *sprint review* é realizado. Segundo [Schwaber 2004] nele é feito uma apresentação ao *Product Owner* de tudo o que foi desenvolvido durante a *sprint*, após isso, é feita a *Sprint Retrospective* onde são feitas as seguintes perguntas ao time:

- O que aconteceu durante a *sprint*?
- O que foi bom durante a *sprint*?
- O que poderia funcionar melhor?

Essa prática deve ser considerada de extrema importância para a equipe se conhecer melhor e melhorar nas próximas *Sprints*.

2.4 Considerações Finais

O surgimento das metodologias ágeis gerou muita polêmica no meio acadêmico. Com o passar do tempo, diversos artigos e estudos foram publicados comparando o modelo tradicional com as metodologias ágeis. Em (Charete, 2001) é possível perceber através dessa comparação que projetos utilizando metodologias ágeis obtiveram melhores resultados em termo de cumprimento de prazos, de custo e de qualidade. Atualmente, a utilização das metodologias ágeis vem crescendo e que dependendo da natureza do projeto podem ser a melhor escolha, como por exemplo projetos onde o cliente não sabe muito bem o que quer no início, ou que tem expectativa de muitas mudanças nos requisitos.

2. Engenharia de Requisitos

Esse capítulo aborda os principais conceitos e etapas que envolvem a Engenharia de Requisitos com enfoque na Priorização de Requisito, especificamente, na metodologia Kano

3.1 Conceituação

Estudos recentes comprovam que os problemas relacionados aos requisitos do sistema afetam várias organizações que desenvolvem sistemas de software (Sommerville, 2007). Portanto, devemos considerar que a Engenharia de Requisitos é uma das fases mais importantes do processo de Engenharia de Software.

A Engenharia de Requisitos (ER) envolve o processo de entendimento e refinamento das necessidades do cliente para um projeto de software, que tem o objetivo de uma correta especificação dos requisitos de software. É a primeira de várias etapas da Engenharia de Software. A principal preocupação da ER envolve o entendimento de quais realmente são requisitos do sistema, como também, a documentação, análise, validação e gerencialmente dos requisitos. É um processo que envolve todas as atividades exigidas para criar e manter o documento de requisitos do sistema.

Existem várias definições na literatura para o termo requisito, mas eles podem descrever, segundo (Kotonya, 1997):

- Uma facilidade no sistema, no nível de usuário, por exemplo, *autocomplete* de uma pesquisa.
- Uma restrição específica do sistema, por exemplo, um *timeout* de resposta de uma operação.
- Uma restrição no desenvolvimento do sistema, por exemplo, uso específico de um banco de dados.

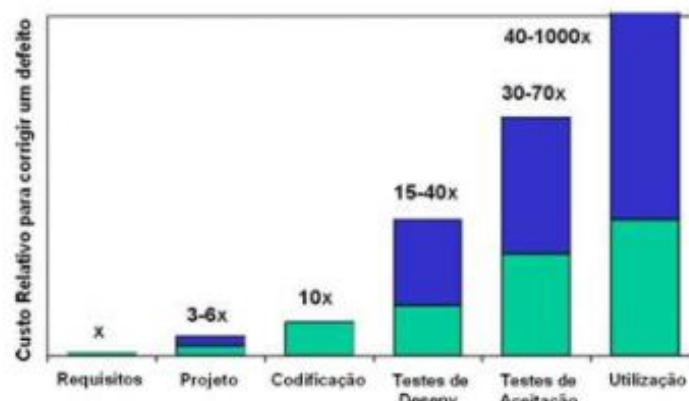
De acordo com (Kotonya, 1997). os requisitos podem ser classificados como:

- Funcionais – capazes de descrever funções e características que o sistema pretende satisfazer, reações a determinadas entradas.
- Não-funcionais – capazes de descrever aspectos relacionados às qualidades associadas ao sistema, como por exemplo, desempenho, usabilidade, segurança.
- Organizacionais – capazes de descrever a estrutura da organização, como metas, filosofia e políticas adotadas por seus funcionários e suas expectativas.

O momento mais crítico do desenvolvimento de software é definir precisamente os requisitos do projeto, permitindo que essa etapa esteja mais suscetível a erros no sistema, não sendo recuperada por nenhuma outra e muito mais difícil de ser posteriormente consertada.

De acordo com a Figura 1, quanto mais tempo passa mais caro fica a correção de um defeito. Por isso, é muito importante que os defeitos sejam encontrados nas primeiras fases do projeto evitando, dessa forma, altos custos para alterações nas fases seguintes

Figura 1. Custo relativo para corrigir um defeito.



Fonte: Kotonya, 1997.

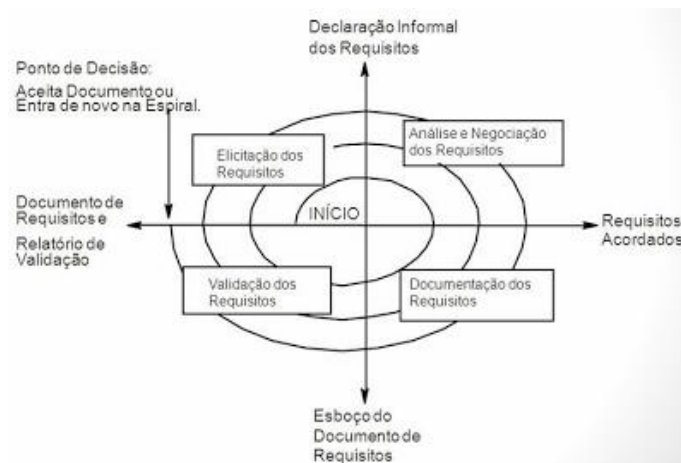
3.2 Etapas da Engenharia de Requisitos

Em (Klaus Pohl, 2010), pode-se considerar que as fases da Engenharia de Requisitos estão divididas da seguinte forma:

- **Elicitação:** atividade de coleta e compreensão de requisitos novos ou existentes junto aos stakeholders.
- **Análise:** processo de análise das necessidades do cliente e usuários para otimização das definições dos requisitos de software.
- **Documentação:** é o processo de elaboração documental e de diagramas do requisitos que foram definidos.
- **Validação:** etapa em que se tenta assegurar que a especificação de requisitos de software estejam de acordo com o que foi elicitado nas primeiras fases.
- **Gerenciamento:** planejamento e controle dos requisitos durante o andamento do projeto.

É possível ver na Figura 2 que todo processo espiral se comporta de forma iterativo incremental. Podemos verificar que cada uma das fases é realizada mais de uma vez até o momento que o documento é considerado aceito. O requisito está apto para ser priorizado a qualquer momento, desde que esteja finalizado. É importante dizer, que mudanças nos requisitos faz parte de todo projeto, devendo ser tratado com naturalidade, principalmente se houver um gerenciamento adequado dessas mudanças. Neste contexto, as metodologias ágeis são consideradas mais adequadas para tratar tais problemas. Enquanto quando se tem um escopo muito bem definido e fechado às metodologias tradicionais são mais indicadas. Nos tópicos a seguir serão apresentadas cada uma das fases do processo de ER

Figura 2. Atividades no modelo espiral do processo de ER



Fonte: Kotonya, 1997.

3.2.1 Elicitação de Requisitos

Etapa inicial do Processo de Engenharia de Requisito, possui a finalidade de esclarecer o conhecimento envolto do problema. Na maioria das vezes, pressupõe uma maior iteração com os stakeholders para realização da coleta dos requisitos. Além de identificar as necessidades dos stakeholders esta atividade requer uma análise minuciosa da organização, do domínio da aplicação e dos processos organizacionais. Como também, (Castro, 2002) sugere que existe a necessidade de capturar as intenções e desejos dos stakeholders. No final do processo de elicitação de requisitos, a quantidade de informação resultante geralmente é alto. Informações essas que precisam ser criteriosamente organizadas, de forma que permita sua compreensão e utilização nas demais etapas da ER.

Várias das abordagens utilizadas no processo de elicitação são oriundas das Ciências Sociais, como por exemplo, Sociologia ou Psicologia onde os analistas precisam entender tais abordagens para seu contexto. Dentre essas técnicas podemos destacar: entrevista, questionário, observação, reuniões em grupo, análise de documentação, etc. Segundo (Patricio, 2004), as importantes contribuições para as elicitações de requisitos são propostas pela área de marketing, uma vez que ela trabalha com aspectos de avaliação e aceitação de produtos os serviços considerado a perspectivas do consumidores ou usuários.

3.2.2 Análise e Negociação dos Requisitos

Fase onde são coletadas as informações sobre requisitos, de forma, que possam ser checadas, discutidas, melhor entendidas, priorizadas e negociadas. Envolve também a discussão dos conflitos existentes entre os requisitos e a busca por uma solução consensual, na tentativa de gerenciar possíveis conflitos e atender as necessidades das partes envolvidas. Os requisitos finais devem refletir um compromisso entre as necessidades estratégicas da organização, dos requisitos específicos de diferentes partes interessadas, das restrições do projeto e limitações de prazo e orçamento. Segundo (Kotonya, 1997), constitui-se as principais atividades da fase de análise de requisitos:

1. Verificação da importância dos requisitos – analisar a real necessidade dos requisitos elicitados, ou seja, se eles encontram-se alinhados às metas e objetivos de negócio da organização, bem como, para o problema ser abordado; esta atividade está diretamente ligada ao foco deste trabalho que trata da priorização usando o método Kano.

2. Verificação da consistência e completude dos requisitos – a averiguação da consistência checando os requisitos para que estes não sejam contraditórios. Já a verificação da completude, compromete-se em examinar se nenhum serviço ou limitação, necessária, foi esquecido.

3. Verificação da viabilidade dos requisitos – o resultado da elicitação é uma grande lista de requisitos. Faz-se necessário confirmar se eles são viáveis em relação a outras variáveis do projeto, como tempo, orçamento, recursos etc.

Para atender as atividades desta fase de forma eficaz são utilizadas as técnicas a seguir:

Matrizes de interação – utilizada para evidenciar a interação entre requisitos e facilitar o processo de análise de possíveis conflitos entre eles. De forma simplificada, para construção dessas matrizes utiliza-se uma tabela com linhas e colunas rotuladas de forma que identificam os requisitos; os valores numéricos

indicam a relação entre cada um dos requisitos, podendo verificar conflitos ou sobreposições.

Lista de verificação (checklists) – corresponde à lista de questões, disponibilizadas para o analista, de forma a auxiliá-lo na avaliação de cada requisito. São extremamente úteis, porque oferecem uma referência sobre o que procurar e reduzem as chances de que requisitos muito importantes deixem de ser analisados. Ao final da verificação, pode-se fornecer uma lista de inconsistências encontradas, as quais podem ser solucionadas por meio de negociação ou nova elicitación de requisito;

Prototipação – os protótipos são criados na etapa de elicitación, podendo ser aperfeiçoados na etapa de análise e negociação, permitindo uma avaliação mais rica dos requisitos do sistema. Contribuem para um maior envolvimento entre as partes interessadas durante as atividades de elicitación, análise e negociação dos requisitos.

Reuniões – pode-se considerar o meio mais eficiente de negociação e resolução de conflitos entre requisitos. Geralmente são conduzidas em três etapas:

- Explicação sobre a natureza dos problemas de requisitos;
- Discussão sobre o melhor modelo de resolver os problemas atentando-se às prioridades estabelecidas;
- Resolução dos conflitos mediante tomada de ações corretivas.

Priorização de requisitos

A priorização de requisitos possui como objetivo sanar conflitos existentes entre as necessidades dos usuários, de uma forma que não impacte na satisfação dos objetivos de cada usuário. De forma geral, os modelos de priorização identificam as necessidades mais importantes de cada usuário, ou seja, atribuem prioridade aos requisitos, logo em seguida, analisa os resultados para tentar priorizar o atendimento

de requisitos mais críticos. As principais vantagens da Priorização de Requisitos incluem:

- Evitar escolhas entre objetivos conflitantes como qualidade, custo e *time-to-market*.
- Priorizar recursos baseados na importância para o objetivo do projeto.

Segundo (Karlsson, 2003), geralmente a priorização é efetuada através de meios informais e pouco efetivos. Uma das dificuldades enfrentadas é a forma como a priorização é realizada, muitas vezes não fica claro para o cliente qual o impacto real que a prioridade de certos requisitos podem trazer para seu negócio. Na prática, muitas empresas de software realizam esse processo informalmente, gerando softwares que não tem tanta qualidade. Mesmo com o progresso da engenharia de requisitos ainda não existe uma forma simples, eficaz e industrialmente aplicada para priorizar requisitos.

3.2.3 Documentação dos Requisitos

Nesta etapa os documentos gerados estão num nível maior de detalhamento, quanto ao escopo dos requisitos, servindo com insumo para o restante do processo de desenvolvimento do software. O documento de requisito é um forma possível de descrever as características do produto e método de desenvolvimento, interface com outras aplicações e informações de suporte ao problema.

Os documentos de requisitos do sistema podem ser apresentados tanto através da linguagem natural, quando da lógica. A escolha da melhor forma deve considerar fundamentalmente a natureza do sistema de software que está sendo desenvolvido, a frequência de mudanças e volatilidade dos requisitos.

O documento de requisitos serve como um contrato entre os analistas e stakeholders, devendo, portanto ser formatado e estruturado de acordo com os padrões organizacionais (Kotonya, 1997). Para uma documentação ser considerada completa e consistente é necessário o rastreamento dos requisitos, que pode se dar de forma horizontal ou vertical. O rastreamento horizontal, quando se faz

relacionamento entre os artefatos de requisitos e o vertical quando se encontra centrado no relacionamento de artefatos de diferentes fases do processo de desenvolvimento do software.

Não pode-se considerar que é uma atividade trivial, visto que, são necessárias ferramentas específicas para suporte do gerenciamento de requisitos no acompanhamento adequado de alterações e evoluções dos requisitos. Depois de produzida a documentação adequada, ela permite a organização e monitoramento de requisitos enquanto auxilia no processo de controle de mudanças dos mesmos, quando necessárias.

3.2.4 Validação dos Requisitos

Podemos considerar a validação de requisitos um processo de certificação da documentação de requisitos, ou seja, para garantir que eles estejam em conformidade com as necessidades dos stakeholders. Essa ótica da atividade de validação retrata um processo contínuo, que na maioria das vezes, ocorre em paralelo com as fases de elicitação e especificação.

Nesta fase lida-se muito com a dificuldade de obtenção de um consenso entre vários stakeholders, que podem possuir objetivos conflitantes, podendo ser prejudicada quando não se existe um comprometimento do stakeholder para uma discussão junto ao analista dos requisitos. Este problema pode ser agravado quando o produto está sendo desenvolvido pela primeira ou quando há um método para demonstrar que uma especificação de requisitos esteja correta em relação a outras representações do sistema. (Kotonya, 1997).

No entanto, devemos destacar que existe uma variedade de técnicas passíveis de ser aplicadas, no apoio ao processo de validação, sendo elas:

- **Prototipação** – se o protótipo foi desenvolvido com finalidade de elicitação de requisitos, é possível utilizá-lo adiante para validação dos requisitos. Porém, para a validação, devem ser mais completos e conter requisitos suficientes que possam garantir as facilidades geradas pelo sistema, esteja de acordo com o

uso prático que é esperado pelo stakeholder. Os protótipos de elicitação, na maioria das vezes não contemplam as funcionalidades e mudanças ocorridas durante o processo de análise de requisitos. Por isso, faz-se necessário dar continuidade ao desenvolvimento do protótipo durante a etapa de validação.

- **Teste de Requisitos** – técnicas baseadas em “cenário”, que permitem elicitar e analisar requisitos enquanto viabilizam a criação de casos de teste para os cenários identificados, (Cysneiros, 2002). A execução dos requisitos implementados pode ser simulada para demonstrar que todos os requisitos do sistema estão sendo considerados de acordo com o esperado. Na presença de dificuldades, quando analisados casos de teste para um dado requisito, isto quer dizer, que pode existir algum problema na definição do requisito.

- **Revisões** – pode ser considerada a técnica mais utilizada. Envolve um grupo de pessoas lendo e analisando a documentação de requisitos em busca de possíveis problemas. É processada da seguinte forma: uma reunião formal, onde o analista de requisitos apresenta cada um dos requisitos para crítica e identificação de inconsistências pelo grupo, após identificação, são registradas, para serem colocadas em discussão posteriormente. O grupo de revisão deve tomar as decisões com a finalidade de corrigir o problema encontrado.

Após finalizar a etapa validação de requisitos, podemos dizer que há um conhecimento detalhado do domínio do problema e dos requisitos importantes do sistema de software a ser desenvolvido, fazendo assim, a entrega do Documento de Requisitos do Sistema. Caso não houver a aceitação na entrega do documento, um novo ciclo deve ser iniciado e qualquer atividade do Processo de Engenharia de Requisitos poderá ser realizada mais uma vez, dependendo da necessidade. Na prática a validação não é só aplicável ao modelo final de requisitos, como também, a todos os modelos intermediários gerados ao longo do processo.

3.2.5 Gerenciamento dos Requisitos

Embora não haja gerenciamento dos requisitos no modelo espiral de Kotonya, esta é uma importante atividade no processo de ER e envolve todas as fases

descritas anteriormente. Consistindo em gerenciar as mudanças dos requisitos propostos, surgindo principalmente, quando há alterações nas prioridades do negócio, quando se identificam erros ou omissões nos requisitos, ou até, quando novos requisitos são definidos. De forma geral o gerenciamento de requisitos envolve os seguintes passos:

- **Identificação dos requisitos** - A identificação do requisito é importante para se obter a referência cruzada dele e para ser feita a avaliação de rastreamento;
- **Gerenciamento de mudanças** – trata-se de um conjunto de atividades que avalia o impacto e o custo da mudança.
- **Políticas de rastreamento** – políticas que definem as relações entre os requisitos, assim como entre os requisitos e o projeto.
- **Suporte de ferramenta CASE** – o gerenciamento dos requisitos pode envolver uma grande quantidade de informações, ou seja, é necessário o auxílio de uma ferramenta para o bom desempenho desta atividade. Essas ferramentas compreendem desde sistemas especializados a planilhas de cálculos ou bancos de dados.

O gerenciamento dos requisitos envolve também o entendimento e controle das mudanças nos requisitos do sistema, (Sommerville, 2007). Não constitui uma atividade trivial, utilizando-se de ferramentas específicas para lidar com as informações e alterações dos requisitos durante sua evolução. Por isso, ressalta-se que a rastreabilidade é um processo eficiente e necessário no gerenciamento dos requisitos.

3.3 Técnicas de Priorização

3.3.1 Visão Geral das Técnicas de Priorização de Requisitos

Existem mais de 50 técnicas de priorização de requisitos, que implementam diferentes estratégias. A Tabela 1 sintetiza algumas das técnicas de priorização mais conhecidas (Junior, 2015). A aplicação ou não de cada uma, assim como o desempenho dela, varia de acordo com o projeto, como por exemplo:

- Qual o grau de clareza dos requisitos?
- Tipo de desenvolvimento, linear ou não?
- Qual o grau de alteração dos requisitos?
- Qual o nível de participação dos *stakeholders*?

Tabela 1. Algumas das principais técnicas de priorização de requisitos.

TÉCNICA	DESCRIÇÃO	COMENTÁRIOS
Analytic Hierarchy Process	Compara todos os possíveis pares de requisitos hierárquicos. Primeiro são identificados os atributos e as alternativas para cada requisito e constrói-se uma hierarquia. Em seguida, são especificadas as preferências para cada par de atributos usando-se uma escala de 1 (para mesma prioridade) até 9 (diferença extrema); depois deriva-se uma prioridade numérica para cada elemento da hierarquia.	Voltada para decisões complexas; consome bastante tempo dado o alto número de comparações. Adequada para modelos de desenvolvimento lineares e em requisitos pouco contraditórios; não é adequada na modelagem iterativa e com requisitos ambíguos.
Ranking	Requisitos mais importantes ranqueados como 1 e os menos importantes com n; Se o número de requisitos for grande, e se desejar um alto grau de exatidão, Bubble Sort e Binary Search Tree são mais adequados porque o Ranking, necessita que o usuário ser recorde de um número maior de requisitos por vez.	Eficaz quando usado o modelo de desenvolvimento linear. Nas outras situações, o seu desempenho é muito menor do que a maioria das outras.
Binary Search Tree	Organiza os requisitos em árvore de modo que onde os requisitos filhos à esquerda do nó têm menos prioridade do que o nó, e requisitos à direita têm maior prioridade do que o nó.	Adequada para quantidade menor ou igual a 15 requisitos, pela sua complexidade em comparar cada nó da árvore. A complexidade desta, é: $O(n \log n)$.
Hundred Dolar (\$100)	Cada stakeholder distribui US \$100 entre os requisitos a serem priorizados; o resultado indica quanto um requisito é mais/menos importante do que outro em termos proporcionais.	Indicada para requisitos bem detalhados e que geram pouca discordância. Funciona muito bem para uma quantidade baixa de requisitos.

Algoritmos Genéticos	Calcula a prioridade dos requisitos através do cruzamento entre: interesse dos stakeholders em cada requisito, dependência dos requisitos e o peso do perfil do stakeholder no projeto.	Consegue lidar com grande número de requisitos. Minimizando as discordâncias entre stakeholders. Trata dependências entre os requisitos
Kano	Esta técnica aborda a classificação dos requisitos levando em conta o sentimento de satisfação do cliente. Inicialmente os requisitos são classificados em: mandatórios, lineares, indiferentes, desejados e reversos.	Para a captação e separação dos requisitos nas categorias propostas, é necessária a aplicação de questionários compostos por um quadro que deve ser preenchido pelos stakeholders.

Fonte: Adaptado de (Junior, 2015).

3.3.2 Priorização utilizando Kano

A técnica de priorização Kano foi desenvolvida por Noriaki Kano em 1984 (Cohn, 2005). É um modelo muito encontrado na área de gerência de qualidade e no uso do *marketing*, para saber as necessidades do cliente e satisfazê-las. Kano é uma técnica para desenvolvimento ou melhoria de produtos baseado na caracterização da necessidade do cliente, sendo elas explicitamente verbalizadas ou não. O diagrama de Kano possibilita aos desenvolvedores de produtos e serviços transformarem as informações obtidas pelas pesquisas de *marketing* em melhorias reais no produto. Essa técnica não possui como objetivo apenas a satisfação do cliente, mas a superação de suas expectativas.

É importante ressaltar que a técnica Kano não foi originada na área da computação, mas sim, na administração. Ela está alinhada com os princípios do manifesto ágil “Colaboração com o cliente ao invés de negociação de contratos”. A técnica utiliza a opinião dos clientes almejando atingir o máximo de participação deles.

A técnica Kano apresenta uma metodologia para priorizar as partes dos produtos em:

1. Indispensável ou obrigatório.
2. Importantes ou linear.
3. Desejáveis.

De acordo com (Cohn, 2005), a divisão citada tem o intuito de priorizar as características que deixam o cliente mais satisfeito. Ou seja, os requisitos indispensáveis devem estar presentes, embora eles não estejam ligados com a uma a uma grande satisfação do usuário.

Os requisitos importantes ou lineares podem ser descritos da seguinte maneira: são aqueles diretamente relacionados com a satisfação e aumentam de forma progressiva. Já os requisitos desejáveis fornecem uma grande satisfação quando estão presentes, mesmo que isso implique numa adição do preço ao produto para satisfazê-lo. A questão principal é que o fato de uma característica não estar presente pode não trazer nenhuma decepção ao cliente, muito menos diminuir a satisfação com o produto final; geralmente, são chamadas de necessidades desconhecidas, pois o cliente nem sequer sabe que precisam delas até possuí-las.

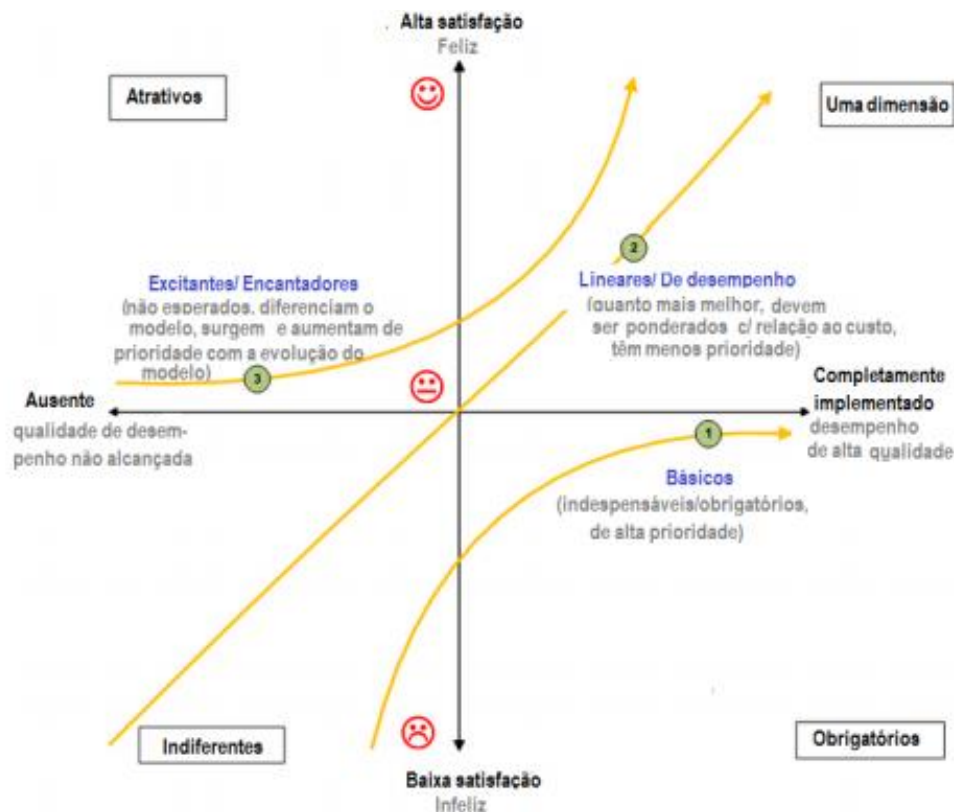
Observando a Figura 3, pode-se verificar que o que trás maior satisfação são os requisitos desejáveis, mas essa satisfação só pode ser alcançada se ao menos os requisitos indispensáveis forem satisfeitos. Por isso, a ordem de priorização dos requisitos é: primeiro são realizados os requisitos indispensáveis, depois os importantes e por fim os desejáveis, isto é, aqueles que trarão um valor agregado muito maior se forem atendidos. No entanto, os outros devem ser atendidos antes uma vez que são necessidades consideradas básicas e que estão associadas diretamente com a valorização do produto.

A técnica Kano consiste em fazer duas perguntas para cada requisito (funcional e disfuncional). Existem outros estudos que utilizam essas perguntas como (Robertson, 2006), da satisfação e insatisfação caso o requisito X ou Y seja implementado. No estudo, essas respostas são colocadas em valores de 1 a 5, sendo o 1 para satisfação, não preocupado se esse requisito vai ser satisfatoriamente implementado e o 5 para muito feliz se esse requisito for satisfatoriamente implementado.

O paralelo feito entre o que deixa o cliente muito satisfeito e o que deixa infeliz se não estiver presente na próxima entrega é de grande importância para se ter a real relação da necessidade de cada requisito. Naturalmente, para cada

requisito vai se atribuir prioridade máxima pelos clientes. Nestes casos, é interessante incluir o critério de insatisfação para perceber o que realmente o deixa infeliz se não for desenvolvido. No geral, as variações de Kano proposta na literatura utilizam os critérios de satisfação e insatisfação que o cliente atribui para cada requisito.

Figura 3. Diagrama de Kano sobre satisfação do cliente.



Fonte: Adaptado de Rozenfeld et al, 2006.

Mas, não necessariamente os requisitos importantes devem ser desenvolvidos no início; porém, eles devem estar concluídos antes do produto final ser entregue, visto que, a satisfação do cliente não será atingida caso falte algum item indispensável.

3.3.3 Como Utilizar Kano

Segundo (Cohn, 2005), Kano é uma técnica fácil de ser aplicada, geralmente demanda pouco tempo para sua utilização. O instrumento de coleta de dados do método Kano é constituído de um questionário no qual, para cada característica a ser avaliada, é elaborado um par de questões que o consumidor poderá responder de cinco maneiras diferentes, sendo uma questão funcional, quando o atributo está presente ou seu desempenho é superior, e a outra disfuncional, formulada com a preocupação da reação dos clientes quando o atributo estiver ausente ou seu desempenho for insuficiente

De acordo com (Cohn, 2005), idealmente para se aplicar a técnica Kano é necessário em torno de 20 a 30 usuários respondendo à pesquisa; as possíveis opções de respostas são:

1. Seria melhor dessa forma
2. Eu esperava dessa forma
3. Estou neutro
4. Eu consigo aceitar dessa forma
5. Não gosto dessa forma

As combinações das respostas às perguntas disfuncionais e funcionais gera um resultado para cada requisito, que determina a importância do mesmo. As perguntas seriam elaboradas da seguinte forma:

- Como você se sentiria caso o Requisito X, estivesse no próximo release?
- Como você se sentiria caso o Requisito X, não estivesse no próximo release?

Figura 4. Resultado das combinações às respostas funcionais e disfuncionais.

		Gostaria	Deveria	Indiferente	Suporta	Não gostaria	
Funcional	Gostaria	Q	D	D	D	L	
	Deveria	R	I	I	I	M	
	Indiferente	R	I	I	I	M	
	Suporta	R	I	I	I	M	
	Não gostaria	R	R	R	R	Q	

Legenda
M - Mandatário
L - Linear
D - Desejado
I - Indiferente
R - Reverso
Q - Questionável

Fonte: Cohn, 2005.

As categorias “mandatário/indispensável”, “importante/linear” e “desejado” serão explicadas na Seção 4.1. O “questionável” se refere a quando o cliente utiliza-se de respostas contraditórias, o que pode significar que ou ele não entendeu as perguntas ou não está respondendo com verdade. O “reverso” significa que se aquele requisito vier a ser desenvolvido, poderá trazer rejeição ao software ou à determinada funcionalidade ao invés de satisfação. O “indiferente” é o que está presente na maior quantidade de combinações e é utilizado quando o usuário demonstra que não tem necessidade para que esta funcionalidade seja desenvolvida.

Após enviar as perguntas aos usuários o resultado deve ser consolidado através dos resultados obtidos na Figura 4 formando-se a Tabela 2.

Tabela 2. Simulação dos resultados obtidos.

Requisito	D	L	M	I	R	Q	Classificação
1	18,4	<u>43,8</u>	22,8	12,8	1,7	0,5	Linear
2	8,3	30,9	<u>54,3</u>	4,2	1,4	0,9	Mandatário
3	<u>39,1</u>	14,8	<u>36,6</u>	8,2	0,2	1,1	Desejado Mandatário

Na Tabela 2, podemos verificar uma situação fictícia onde os usuários responderam para os três requisitos presentes suas respectivas classificações. No Requisito 3, apesar da maioria ter considerado o requisito **mandatório**, muitos também o consideraram **desejado**, por isso este requisito foi classificado como **mandatório** e **desejado**. Na forma original de Kano, uma vez obtida a tabela disposta na Tabela 2, já se é suficiente para iniciar a priorização de requisitos. Na próxima seção tentaremos adaptar esse processo para melhorar sua visualização e conseqüentemente facilitar na tomada de decisão de quais requisitos priorizar.

3.4 Considerações Finais

A Engenharia de Requisitos é composta por vários processos que envolvem vários *stakeholders*. Apresentamos nesse capítulo as principais etapas do processo de requisitos. Em geral, pode-se considerar um processo crítico para sistemas que precisam ter um curto *time-to-market* e em contextos bastante competitivos. No próximo capítulo é apresentada uma proposta para priorizar requisitos em projetos ágeis.

Existem diversas técnicas de priorização. A técnica destacada neste trabalho é a Kano, uma técnica simples baseada na caracterização da necessidade do cliente, sendo elas explicitamente verbalizadas ou não; essa técnica não possui como objetivo apenas a satisfação do cliente, mas a superação de suas expectativas.

4. Priorização de Requisitos em Ambientes Ágeis

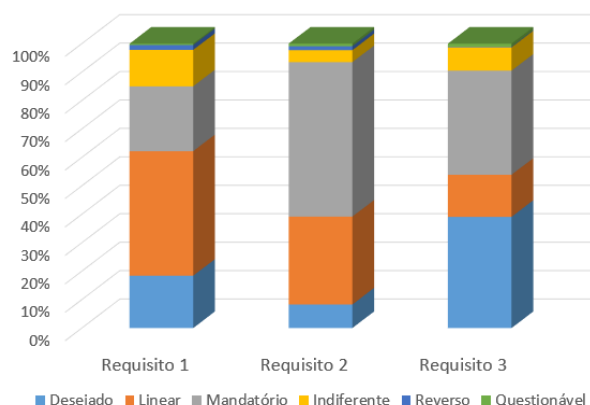
Este capítulo possui como finalidade apresentar uma proposta de priorização de requisitos para projetos de desenvolvimento em ambientes ágeis.

4.1 Novos elementos visuais para Kano

Kano é muito interessante quando se trata do conceito e dos tipos de dados obtidos. Porém, com relação à visualização dos dados dela podemos dizer que é bem insatisfatória, visto que, torna-se muito abstrato distinguir quais requisitos vão ser priorizados. Por exemplo, na Tabela 2 que encontra-se na seção 3.3.3, como saberemos se devemos implementar o requisito 2 ou 3, visto que ambos são mandatórios? Em ambientes ágeis, devemos considerar que as tomadas de decisões devem se dar de forma rápida; porém, pode ser que a equipe não compreenda a essência dos resultados obtidos a partir da técnica Kano.

O *Scrum* utiliza-se do valor de negócio definido pelo *Product Owner* (PO) e o tamanho definido pelo time durante o *Sprint Planning*. O PO pode ter dificuldade de dizer qual o valor de negócio dos itens se utilizando apenas de sua visão pessoal. Para isso, podemos dispor as informações conforme apresentado na Figura 5.

Figura 5. Gráfico em forma de colunas da Tabela 2



Na Figura 5 é possível perceber com mais clareza quais as classificações distribuídas para os respectivos requisitos. Podemos realizar várias análises através dele, por exemplo, a quantidade de indiferente, reverso e questionável é desprezível em relação aos demais itens. Na figura 5, ainda podemos observar que o Requisito 2 possui o maior quantidade de mandatório, tornando-o um grande candidato a ser implementado primeiro, enquanto que o Requisito 3 mostra que muitos usuários o escolheram como desejável. O Requisito 1, mesmo tendo a maioria dos usuários o classificando como linear pode ser realizado antes do que o Requisito 3. Na Figura 6, podemos visualizar a proporção alta de classificações como Linear (44%). Na Figura 7, visualizamos a disparidade de opção por Mandatório, o que reforça a ideia de que esse deveria ser o primeiro requisito a ser implementado. Já na Figura 8, para o Requisito 3, podemos visualizar a proximidade clara entre Desejado e Mandatório, o que esse alto grau de desejado fez com que o Requisito 1 fosse considerado mais prioritário. Dessa forma a priorização sugerida da seguinte ordem: Requisito 2, Requisito 1 e Requisito 3.

Figura 6. Visualização do gráfico de pizza do Requisito 1

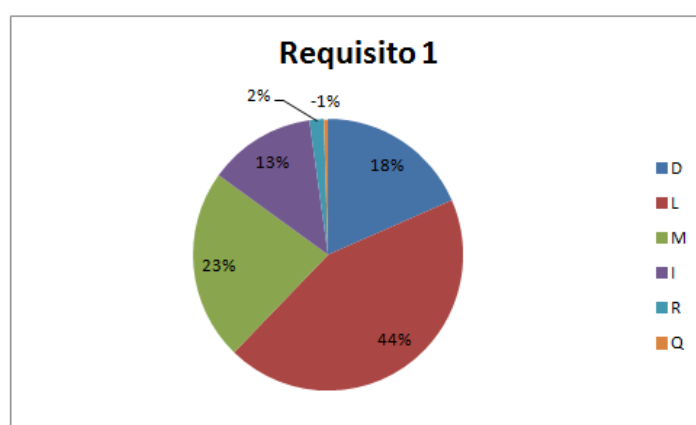


Figura 7. Visualização do gráfico pizza do Requisito 2

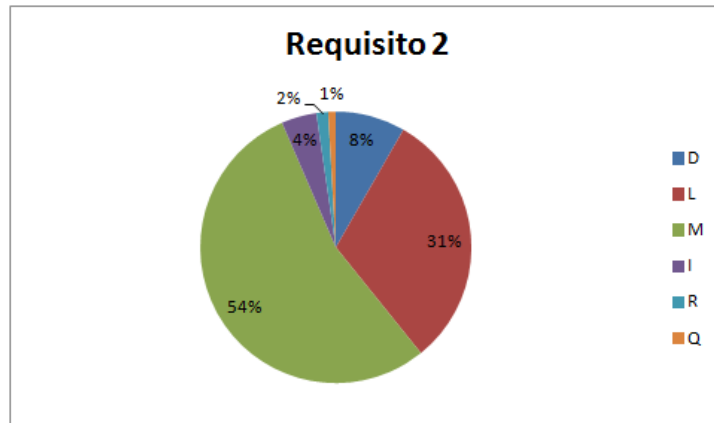
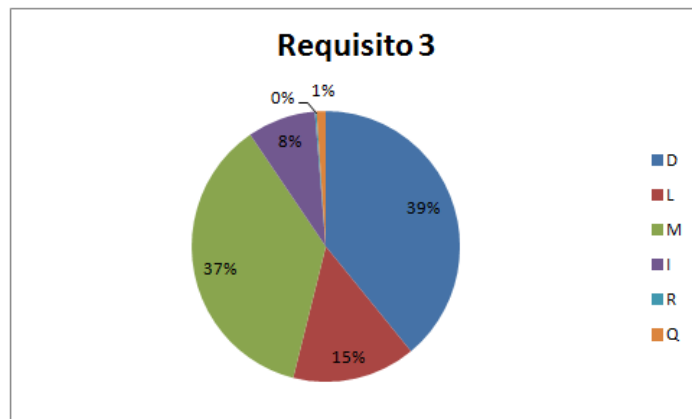


Figura 8. Visualização do gráfico pizza do Requisito 3



4.2 Etapas da Priorização

A partir de agora, iremos apresentar etapas para realização do processo de priorização em ambientes ágeis. Onde utilizando-se do método *Scrum* para detalhar o processo de priorização, justificada pelo fato do *Scrum* utilizar-se de um enfoque maior de atividades de gerências de projetos. Na Figura 9, é apresentada uma visão geral das fases do processo de priorização proposto adicionado das fases já existentes na técnica Kano (Montar Questionário, Aplicar Perguntas, Consolidar Resultados).

Figura 9. Fases da proposta de priorização de requisitos



Etapa 1 – Selecionar Backlog

Como primeiro passo, é necessário selecionar os requisitos que serão priorizados. Idealmente, considera-se até 15 requisitos para o questionário não ser extenso e cansativo para que o usuário possa responder com total atenção todas as perguntas propostas, não impactando na aplicação da técnica.

Etapa 2 – Montar Questionário

A partir de agora, os requisitos escolhidos serão utilizados na montagem das perguntas. As perguntas devem estar dispostas da seguinte forma:

- 1) Se a próxima versão do [Projeto] incluir [Nome do requisito], como você se sentirá? (Pergunta funcional).
- 2) Se a próxima versão do [Projeto] **não** incluir [Nome do requisito], como você se sentirá? (Pergunta disfuncional)

Figura 10. Opções de respostas ao questionário



Etapa 3 – Aplicar Perguntas

Nessa etapa as perguntas que foram montadas são enviadas aos usuários. É necessário explicar que essas perguntas e respostas serão utilizadas na avaliação dos requisitos que serão incluídos na próxima entrega. Esta é considerada uma das fases mais trabalhosas do processo, pois é necessário que todos os usuários participantes respondam ao questionário.

Etapa 4 – Consolidar Resultados

É importante na etapa de consolidação dos resultados a utilização de uma planilha eletrônica para gerar os gráficos com facilidade. O Apêndice A possui uma cópia dessa planilha. Para a entrada de dados dessa tabela, é necessário uma primeira consolidação feita através da Figura 4, que mostra a relação das perguntas com o resultado das combinação. Momento este, onde cada requisito é analisado levando em consideração as duas respostas do usuário, a partir de então sua classificação é feita.

Como podemos observar na Figura 12, caso o usuário responda para uma pergunta Funcional como “Gostaria” e para a pergunta Disfuncional, como “Indiferente”, após realizar cada associação para cada requisito e colocar na planilha como exemplificada no Apêndice A, tem-se os gráficos e percentuais para cada requisito.

Figura 11. Utilização do gráfico para análise

		Disfuncional				
		Gostaria	Deveria	Indiferente	Suporta	Não gostaria
Funcional	Gostaria	Q	D	D	D	L
	Deveria	R	I	I	I	M
	Indiferente	R	I	I	I	M
	Suporta	R	I	I	I	M
	Não gostaria	R	R	R	R	Q

Legenda

- M - Mandatório
- L - Linear
- D - Desejado
- I - Indiferente
- R - Reverso
- Q - Questionável

Etapa 5 – Análise e Interpretação dos Resultados

Através da leitura dos gráficos é possível fazer várias análises. Para realizar a interpretação dos resultados é necessário conhecer as particularidades dos requisitos. Geralmente, deve-se avaliar o requisito pela maioria das classificações, mas outros pontos devem ser analisados como indicadores importantes, como:

- O grau de rejeição daquele requisito. Além de não deixar o usuário contente, pode deixá-lo insatisfeito caso seja implementado.
- O número de “Indiferente” elevado pode mostrar que aquele requisito não tem uma importância tão grande se for implementado, o que nos leva a crer que ele não foi bem pensado, ou o usuário não entendeu o que ele faria.
- Um número grande de “Deveria” aliado a uma grande quantidade de “Gostaria” indica que aquele requisito para alguns é mandatório e importante para outros.
- Dentro das avaliações deve-se descartar respostas pouco expressivas, ou seja, com um índice baixo, geralmente significa que o usuário simplesmente não entendeu a resposta ou até mesmo é contra aquela ideia.

Etapa 6 – *Ranking* de requisitos priorizados

É importante, após toda a análise, elaborar um *ranking* dos requisitos priorizados. Para aqueles que não atingiram um grau de satisfação esperado, deve-se colocá-los em espera;

4.3 Estudo de Caso

Essa seção tem como objetivo apresentar uma aplicação da proposta desse estudo em uma empresa de desenvolvimento de *software* incluída no mercado de Recife avaliando os benefícios da técnica com relação a técnica que já é utilizada pela empresa atualmente.

4.3.1 Aplicação da Técnica Kano

A pesquisa que será apresentada a seguir, foi feita em uma empresa do mercado de desenvolvimento de software do Recife, em um projeto que trabalha com análises de créditos de pessoas físicas. Na pesquisa realizada foi utilizada uma planilha, como exemplificada no Apêndice A; esse arquivo foi enviado por *email* para todos usuários, visto que, os usuários residem em outros estados, dificultando que essa entrega fosse realizada em questionários de papel.

Etapa 1 – Selecionar *backlog*




Na empresa esse trabalho foi realizado pelo P.O. juntamente com o Engenheiro de Requisitos, que é a pessoa responsável pela área. Observaram os itens que estavam em *backlog* no momento, esses itens não haviam sido priorizados pois a cada nova iteração novos pedidos mais urgentes surgiam, dessa forma foram selecionados 8 requisitos.

Etapa 2 – Montagem das Perguntas

O questionário foi elaborado através de uma planilha Excel, como é possível ver na Figura 13, onde os usuários marcavam um “X” em cada resposta para cada requisito, tanto para as perguntas funcionais quanto disfuncionais, ou seja, as perguntas estão dispostas e divididas em dois blocos onde a primeira pergunta é “Se a próxima versão do [Projeto] incluir o requisito abaixo, como você se sentirá?” e outro bloco com a segunda

pergunta “Se a próxima versão do [Projeto] não incluir o requisito abaixo,

Figura 12. Exemplo de questionário aplicado aos usuários.

EMPRESA - PROJETO - ITERAÇÃO					
Pesquisa de avaliação dos requisitos para próxima entrega					
Se a próxima versão do [Projeto] incluir o requisito abaixo, como você se sentirá?					
	Gostaria	Deveria	Indiferente	Suporta	Não gostaria
REQUISITO 1					
REQUISITO 2					
REQUISITO 3					
REQUISITO 4					
REQUISITO 5					
REQUISITO 6					
REQUISITO 7					
REQUISITO 8					
Se a próxima versão do [Projeto] incluir o requisito abaixo, como você se sentirá?					
	Gostaria	Deveria	Indiferente	Suporta	Não gostaria
REQUISITO 1					
REQUISITO 2					
REQUISITO 3					
REQUISITO 4					
REQUISITO 5					
REQUISITO 6					
REQUISITO 7					
REQUISITO 8					

Etapa 3 – Aplicação das perguntas

O questionário montado, ver Figura 13, foi enviado por *email* para os usuários do projeto. Então foram preenchidas com um X as respostas para cada requisito e os resultados foram retornados para o P.O. que iria consolidar esses dados.

Etapa 4 – Consolidação dos Resultados

Nessa fase os dados retornados pelos usuários foram consolidados utilizando o gráfico matriz de Kano. Como podemos observar na Figura 11.

A partir do diagrama da Figura 14, cada resposta do usuário foi comparada ao diagrama, gerando assim, a Tabela 3, onde foram dispostos os resultados finais da integração das perguntas funcionais e disfuncionais.

Tabela 3. Resposta obtida dos usuários

	USUÁRIO 1	USUÁRIO 2	USUÁRIO 3	USUÁRIO 4	USUÁRIO 5	USUÁRIO 6	USUÁRIO 7
REQUISITO 1	Indiferente	Reverso	Desejado	Indiferente	Reverso	Indiferente	Indiferente
REQUISITO 2	Indiferente	Indiferente	Indiferente	Reverso	Mandatário	Indiferente	Indiferente
REQUISITO 3	Linear	Linear	Desejado	Mandatário	Mandatário	Reverso	Indiferente
REQUISITO 4	Desejado	Indiferente	Indiferente	Indiferente	Indiferente	Desejado	Linear
REQUISITO 5	Mandatário	Indiferente	Linear	Mandatário	Mandatário	Indiferente	Indiferente
REQUISITO 6	Indiferente	Indiferente	Reverso	Indiferente	Mandatário	Reverso	Indiferente
REQUISITO 7	Indiferente	Desejado	Desejado	Desejado	Desejado	Indiferente	Indiferente
REQUISITO 8	Reverso	Indiferente	Desejado	Desejado	Linear	Linear	Desejado

Na Tabela 4, foi calculado o somatório das respostas dos usuários de cada requisito. A quantidade de respostas de cada requisito, para cada possibilidade de resposta.

Tabela 4. Somatório das respostas dos usuários classificada por requisito.

	MANDATÓRIO	LINEAR	DESEJADO	INDIFERENTE	REVERSO	QUESTIONÁVEL
REQUISITO 1	0	0	1	4	2	0
REQUISITO 2	1	0	0	5	1	0
REQUISITO 3	2	2	1	1	1	0
REQUISITO 4	0	1	2	4	0	0
REQUISITO 5	3	1	0	3	0	0
REQUISITO 6	1	0	0	4	2	0
REQUISITO 7	0	0	4	3	0	0
REQUISITO 8	0	2	3	1	1	0

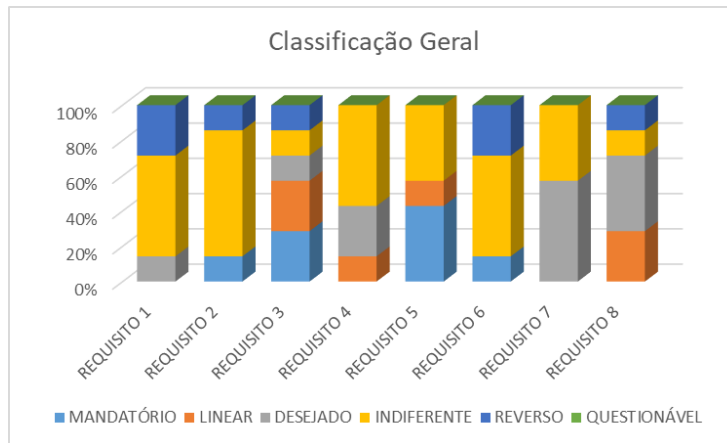
Como próxima etapa, calculamos o percentual de respostas para cada opção, por requisito. Por exemplo, o Requisito 1 obteve 7 respostas onde 4 foram indiferentes, totalizando um percentual equivalente a 57% e dessa forma é montada a Tabela 5, exemplificada abaixo.

Tabela 5. Percentual de respostas obtidas por requisito.

	MANDATÓRIO	LINEAR	DESEJADO	INDIFERENTE	REVERSO	QUESTIONÁVEL
REQUISITO 1	0	0	14,28571429	57,14285714	28,57142857	0
REQUISITO 2	14,28571429	0	0	71,42857143	14,28571429	0
REQUISITO 3	28,57142857	28,57142857	14,28571429	14,28571429	14,28571429	0
REQUISITO 4	0	14,28571429	28,57142857	57,14285714	0	0
REQUISITO 5	42,85714286	14,28571429	0	42,85714286	0	0
REQUISITO 6	14,28571429	0	0	57,14285714	28,57142857	0
REQUISITO 7	0	0	57,14285714	42,85714286	0	0
REQUISITO 8	0	28,57142857	42,85714286	14,28571429	14,28571429	0

Após isso, convertemos esses dados em gráficos de colunas, utilizando o Excel, como na Figura 15.

Figura 13. Visualização dos dados obtidos no gráfico por colunas.



Etapa 5 – Análise e Interpretação dos Resultados

Nessa etapa, os dados consolidados obtidos na fase anterior, são utilizados para realizar análises. Nos gráficos pizzas seguintes podemos observar de forma isolada, para cada requisito, sua classificação proporcional em relação aos critérios.

Tanto o Requisito 1 é um forte candidato a não ser implementado na próxima iteração, pois possui uma grande quantidade de usuários que o definiram como “Indiferente”, além de possuir um grande índice de “Reverso” e não houve nenhuma classificação indispensável, o que o torna um candidato a não ser implementado na próxima iteração. Ver Figura 16.

Figura 14. Gráfico em pizza (Requisito 1)

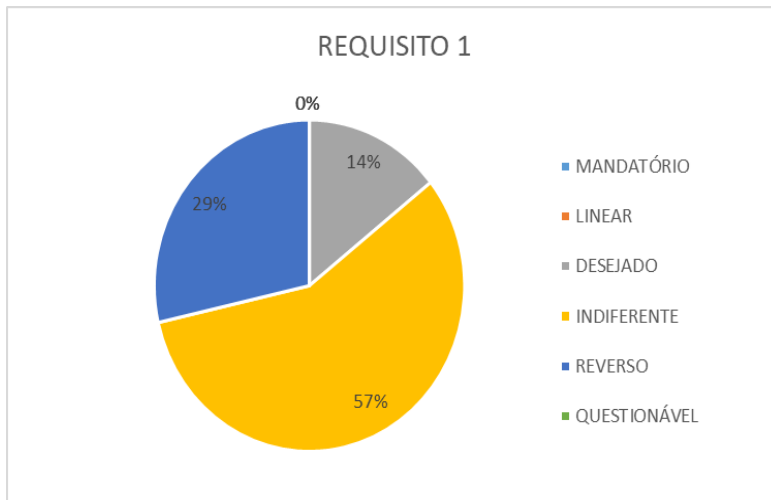
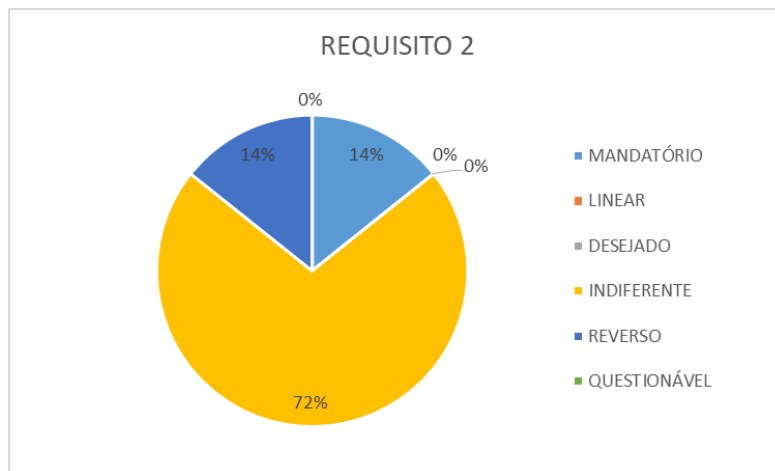


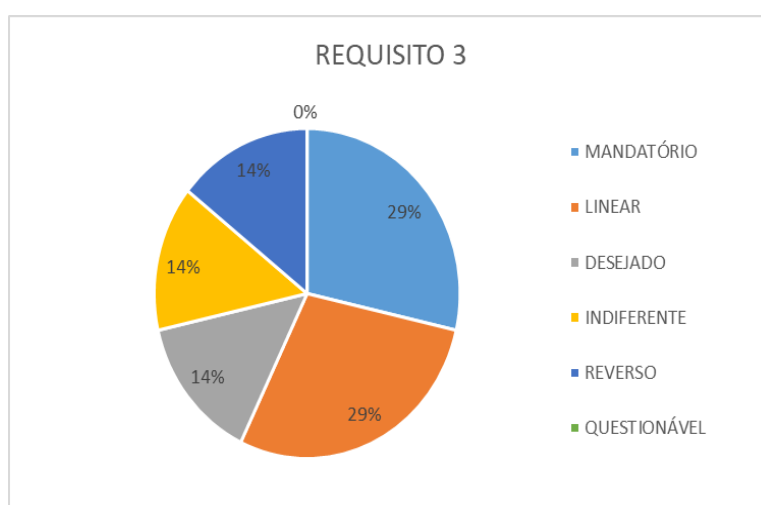
Figura 15. Gráfico em pizza (Requisito 2)



De modo semelhante encontra-se o Requisito 2, ver Figura 17. Porém com a ressalva de que existe 14% dos usuários que o consideram “Mandatário” e “Desejado”, o aconselhado para esse caso é colocar ele em espera uma vez que pode causar satisfação de alguns e insatisfação de outros, o que leva a crer que com um pouco mais de tempo esse requisito passe por uma reformulação e o problema seja sanado.

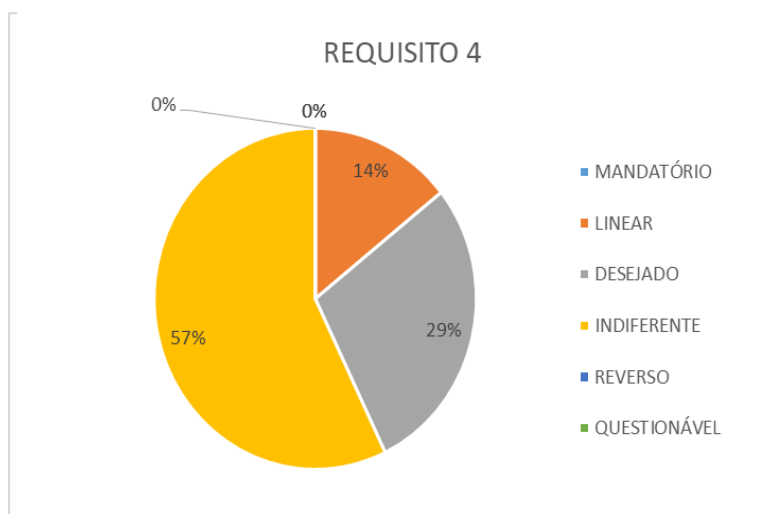
Já o Requisito 3, é claramente muito importante de ser implementado, visto que, os usuários o consideram 29% “Linear”, 29% “Mandatário” e 14% “Desejado”, o que nos faz acreditar que apesar de haver 14% que o consideram “Reverso”, o satisfação seria muito maior do que a insatisfação, caso ele fosse implementado. Ver Figura 18. Assim, o Requisito 3 é colocado na lista de requisitos a serem implementados.

Figura 16. Gráfico em pizza (Requisito 3)



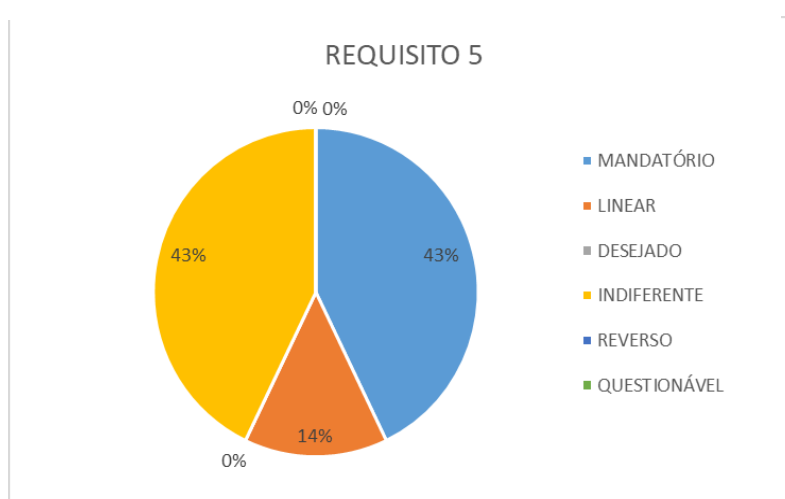
A avaliação que podemos fazer do Requisito 4, é que ele pode ser implementado, por não causar nenhuma insatisfação nos usuários, pelo contrário, possui uma taxa de “Desejado” como 29%, porém por não ter taxa de classificação “Mandatário” sua prioridade deve ser a menor do que o Requisito 3. Ver Figura 19. Assim, resulta por enquanto na seguinte ordem de requisitos priorizados: Requisito 3, Requisito 4.

Figura 17.Gráfico em pizza (Requisito 4)



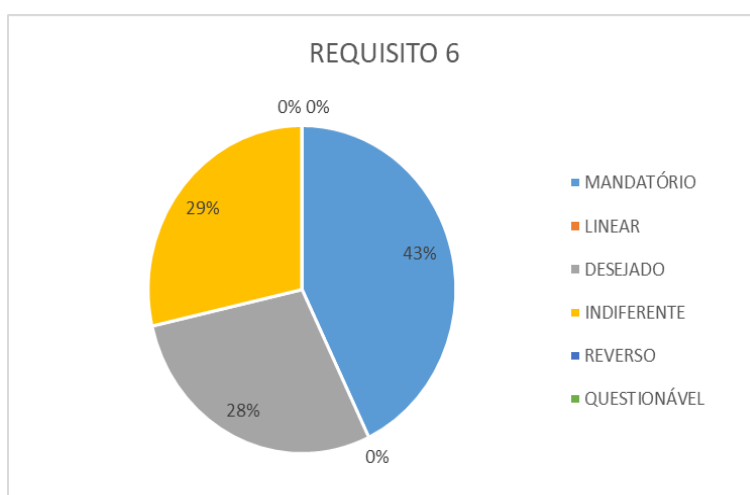
Sobre o Requisito 5, podemos inseri-lo na lista de requisitos priorizados, visto que, possui uma grande quantidade de “Linear” e “Mandatário”, ficando inclusive na frente do Requisito 3, por possuir uma quantidade de classificação obrigatória maior. Ver Figura 20. Resultando, por enquanto, na seguinte ordem de requisitos priorizados: Requisito 5, Requisito 3 e Requisito 4.

Figura 18. Gráfico em pizza (Requisito 5)



Em relação ao Requisito 6, ver Figura 21, podemos considerar que pela grande quantidade de “Mandatório” e “Desejado”, totalizando 71% das respostas o torna um forte candidato a ser implementado, considerando inclusive, que sua prioridade é o maior de todos até agora, o colocando no primeiro da lista, inclusive do Requisito 5, o que apesar de ter a mesma quantidade de “Mandatório”, por não possuir uma quantidade de “Desejado” tão grande quanto o Requisito 6, fica com uma prioridade menor. Tornando a lista de requisitos priorizados com a seguinte ordem: Requisito 6, Requisito 5, Requisito 3 e Requisito 4.

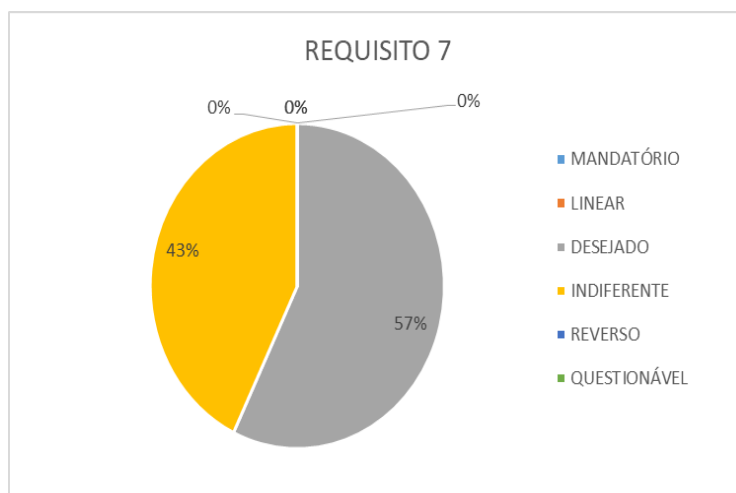
Figura 19. Gráfico em pizza (Requito 6)



Em relação ao Requisito 7, ver Figura 22, nada o impede de ser implementado, pelo contrário, sua implementação pode trazer uma grande satisfação para os usuários, visto que 57% desejam esse requisito na próxima entrega, o que o torna inclusive com prioridade maior do que o Requisito 4, por possuir uma quantidade de “Desejado” maior e “Indiferente” menor, pois o Requisito 4, mesmo somando os seus índices “Linear” e “Desejado” não ultrapassa o nível de “Desejado” do Requisito 7. Porém, ele fica abaixo do Requisito 3, com relação a ordem de priorização, por não

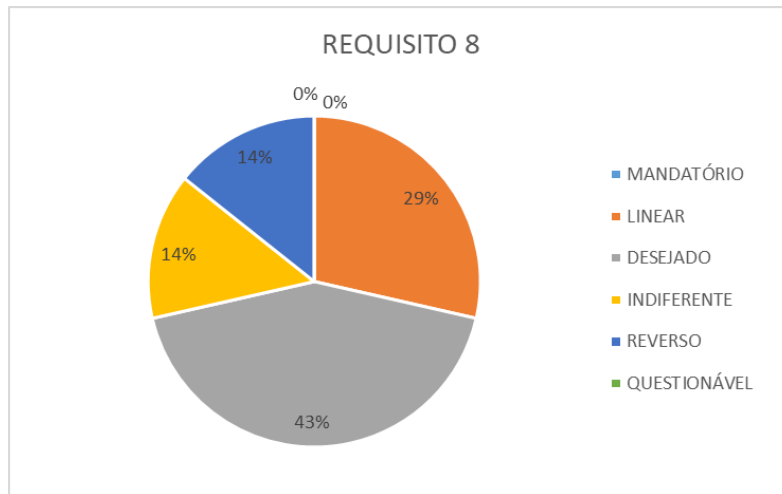
possuir nenhum índice de “Mandatário”. Tornando a lista de requisitos priorizados com a seguinte ordem: Requisito 6, Requisito 5, Requisito 3, Requisito 7 e Requisito 4.

Figura 22. Gráfico em pizza (Requisito 7)



Por fim, o Requisito 8, possui um bom índice de “Linear” e “Desejado” porém, possui 14% do índice definido como “Reverso”, o que o torna semelhante ao Requisito 3, nesse sentido. Porém, o Requisito 8, deve possuir prioridade menor do que o Requisito 4, pois não possui índices “Mandatário” maior. Tornando a lista de requisitos priorizados com a seguinte ordem: Requisito 6, Requisito 5, Requisito 3, Requisito 7, Requisito 4 e Requisito 8.

Figura 20. Gráfico em pizza (Requisito 8)



Etapa 5 – Ranking de Requisitos Priorizados

Nessa fase, após todas as análises realizadas, uma tabela é montada para visualização do *Ranking* dos requisitos, como ilustrado na Tabela 6.

Prioridade	Requisito	Em espera
1	Requisito 6	Requisito 1
2	Requisito 5	Requisito 2
3	Requisito 3	
4	Requisito 7	
5	Requisito 4	
	Requisito 8	

Tabela 6. Sugestão de priorização para próxima entrega.

4.4 Considerações Finais

Uma das principais contribuições deste trabalho encontra-se, principalmente, na estruturação proposta para a implementação da técnica Kano, através de gráficos e análises que podem ser obtidas a partir deles. O

objetivo é proporcionar uma melhor visualização gráfica dos requisitos que necessitam ser priorizados, de acordo com as respostas obtidas pelos usuários.

A pesquisa proposta em Kano deve acontecer durante toda uma iteração, ou seja, isso oferece ao P.O um tempo razoável para aplicação de questionários e toda a consolidação dos dados. Vai servir de insumo para o planejamento da próxima iteração. No caso do *Scrum*, seria o *Sprint Planning*.

A estruturação proposta pode ser promissora em projetos com vários clientes, facilitando que requisitos com maior retorno do produto sejam desenvolvidos antes dos outros, beneficiando os seus clientes.

Durante o estudo de caso algumas dificuldades foram encontradas, tais como: a dificuldade em convencer os gestores da empresa a realizar a pesquisa, a demora na definição dos requisitos do *backlog* e resposta dos usuários. Foi necessário garantir também que o trabalho não violaria os estatutos de confidencialidade impostos pela empresa. Alguns usuários sugeriram a criação de uma ferramenta com uma interface gráfica, onde pudessem acessar para responder o questionário.

O processo ajudou ao P.O. compreender as vontades dos usuários, visto que a técnica atualmente utilizada pela empresa não faz uma avaliação de satisfação do usuário, em relação aos requisitos. Outro ponto positivo, os itens do que encontravam-se no *backlog* estavam a muito tempo sem análise, a proposta forçou P.O. em avaliar esses itens. A aceitação foi bem positiva em relação aos usuários, apesar da proposta esbarrar em algumas resistências por parte da gerência.

5. Conclusão e trabalhos futuros

Nesse capítulo são apresentadas as contribuições desta monografia de conclusão de curso. Além disso, algumas sugestões de trabalhos futuros são propostas.

Este trabalho possui como principal contribuição, uma proposta de um processo de priorização de requisitos para projetos ágeis, baseando-se na técnica Kano e sendo adaptado às necessidades de projetos ágeis, com enfoque no *Scrum*. Foi realizado um estudo de caso em empresa local do mercado de TI, para avaliar a adequação do processo proposto e obtenção de *feedback* dos participantes da pesquisa, no intuito de conduzir melhorias futuras. Tendo em vista que a técnica utilizada para priorização de requisitos atualmente envolve uma curva de aprendizado muito grande, onde apenas um Engenheiro de Requisitos possui acesso e conhecimento sobre a utilização dela dentro da empresa, com a proposta desse trabalho a priorização pode se tornar algo mais iterativo e utilizar isso em benefício da empresa.

Em relação aos trabalhos futuros, foi sugerido pelos usuários a criação de uma ferramenta *web*, onde fosse capaz de se adequar a vários projetos e inclusive inserir componentes de *gamificação* onde fosse estimulado os usuários responderem os questionários de forma mais rápida e constante, e a partir dela, se obter o *ranking* de requisitos priorizados.

Bibliografia

Asfora, Diego Maciel. **Uma abordagem para priorização de requisitos em ambientes ágeis**. Dissertação de Mestrado, Recife: Programa de Pós-Graduação em Engenharia da Computação/ Universidade de Pernambuco, 2015.

Carlshamre, P. A **Usability Perspective on Requirements Engineering – From Methodology to Product Development**. Dissertation No. 726 – Linköpings Universitet, Linköping Studies in Science and Technology, Department of Computer and Information Science, Sweden, 2001.

Castro, J. Kolp, M. e Mylopoulos, J. **"Towards Requerements-Driven Information System Engineering: The Tropos Project"**, Information System, Elsevier, Amsterdam, The Netherlands, 2002.

Cockburn, Alistair. **Agile software development**. 1 ed. Boston: Addison-Wesley, 2002.

Cohn, Mike. **Agile estimating and planning**. 1ed. Pearson Education, 2005.

Cysneiros, G. 2002. **"Ferramenta para Suporte do Mapeamento da Modelagem Organizacional em i* para UML"** Recife: Centro de Informática, Universidade Federal de Pernambuco, Tese de Mestrado, 2002.

James A. Highsmith. **Agile software development ecosystems**. 1 ed. Boston: Addison-Wesley, 2002.

Johnson, Jim. **"ROI, it's your job"** Published Keynote Third International Conference on Extreme Programming, Algero, Italy, Maio 26-29, 2002.

Júnior, Jose Maurício Silva, **G-4REPrioritization: Um Guia para Apoio à Escolha de Técnicas de Priorização de Requisitos**. Dissertação de

Mestrado, Recife: Centro de Informática, Universidade Federal de Pernambuco, Tese de Mestrado, 2009.

K. Pohl, C. Rupp. **Fundamentos da Engenharia de Requisitos**, 1ed, vol 3, Santa Barbara, CA, 2010.

Karlsson, L. **Improving Requirements Selection Quality in Market-Driven Software Development**. PhD Thesis. Department of Communication Systems, Lund Institute of Technology, 2003.

Kotonya, G e Sommerville, I. **Requirements Engineering - Processes and Techniques**, John Willy & Sons, 1997.

Agile Manifesto web site. Disponível em: <<http://agilemanifesto.org>>. Acesso em 14 de setembro 2017.

Lakatos, E. Maria; MARCONI, M. de Andrade. **Fundamentos de metodologia científica: Técnicas de pesquisa**. 7 ed. – São Paulo: Atlas, 2003.

Lamsweerde, A. **Requirements Engineering in the Year 00: A Research Perspective**. 22nd. Proceedings of International Conference on Software Engineering. Limerick, Ireland. Jun, 2000.

Palmer, Stefen R. e Felsing John M, A. **Practical Guide to Feature Driven Development**, ed. Prentice Hall. PRT, 2002.

Patrício, L. Cunha, JF. Fisk, RP, e Nunes, NJ. **Customer Experience Requirement for Multi-Plataform Service Interaction: Bringing Services Marketing to the Elicitation User Requirements**. Proceeding of 12th IEEE International Requirimens Engineering Conference, set 2004.

Pressman, R. **Engenharia de Software**. Mc-Graw-Hill, 2011.

Henrique Rozenfeld, Fernando Antônio Forcellini, Daniel Capaldo Amaral, José Carlos De Toledo, Sergio Luis Da Silva, Dário Henrique Alliprandini,

Régis Kovacs Scalice, **Gestão de Desenvolvimento de Produtos: Uma Referência para a Melhoria do Processo**, 1 ed., Saraiva, 2006

Rational Unified Process. Versão 2002 05 00.

Ken Schawaber: **Agile Software Development with Scrum**. Ed. Microsoft Press 2004.

Sommerville, I. **Engenharia de Software**, Pearson Addison-Wesley, 8a. Edição, 2007.

Teles, Vinicius Manhães. **Extreme Programming: aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade**. 1ed. São Paulo: Novatec, 2004.

Weinberg, G. ***The psychology of computer programming***. 1ed. New York: Van Nostrand Reinhold Company, 1971.

Vestola, M. A **Comparison of nine basic techniques for requirements prioritization**. Helsinki University of Technology, 2011.

Apêndice

O apêndice apresenta o artefato utilizado no processo de priorização proposta nesse trabalho. Foi usada uma planilha Excel para ajudar na consolidação dos dados. A planilha apresenta os resultados aplicados no estudo de caso.

	USUÁRIO 1	USUÁRIO 2	USUÁRIO 3	USUÁRIO 4	USUÁRIO 5	USUÁRIO 6	USUÁRIO 7
2	Indiferente	Reverso	Desajado	Indiferente	Reverso	Indiferente	Indiferente
3	Indiferente	Indiferente	Indiferente	Reverso	Mandatório	Indiferente	Indiferente
4	Indiferente	Indiferente	Indiferente	Reverso	Mandatório	Indiferente	Indiferente
5	Linear	Linear	Desajado	Mandatório	Mandatório	Reverso	Indiferente
6	Desajado	Indiferente	Indiferente	Indiferente	Indiferente	Desajado	Linear
7	Mandatório	Indiferente	Linear	Mandatório	Mandatório	Indiferente	Indiferente
8	Desajado	Desajado	Mandatório	Indiferente	Mandatório	Mandatório	Indiferente
9	Indiferente	Desajado	Desajado	Desajado	Desajado	Indiferente	Indiferente
10	Reverso	Indiferente	Desajado	Desajado	Linear	Linear	Desajado
11							
12							
13							
14							
15	MANDATÓRIO	LINEAR	DESAJADO	INDIFERENTE	REVERSO	QUESTIONÁVEL	
16	0	0	1	4	2	0	
17	1	0	0	5	1	0	
18	2	2	1	1	1	0	
19	0	1	2	4	0	0	
20	3	1	0	3	0	0	
21	3	0	2	2	0	0	
22	0	0	4	3	0	0	
23	0	2	3	1	1	0	
24	9	6	13	23	5	0	
25							
26							
27							
28							
29							

	MANDATÓRIO	LINEAR	DESAJADO	INDIFERENTE	REVERSO	QUESTIONÁVEL
REQUISITO 1	0	0	14.28571429	57.14285714	28.57142857	0
REQUISITO 2	14.28571429	0	0	71.42857143	14.28571429	0
REQUISITO 3	28.57142857	28.57142857	14.28571429	14.28571429	14.28571429	0
REQUISITO 4	0	14.28571429	28.57142857	57.14285714	0	0
REQUISITO 5	42.85714286	14.28571429	0	42.85714286	0	0
REQUISITO 6	42.85714286	0	28.57142857	28.57142857	0	0
REQUISITO 7	0	0	57.14285714	42.85714286	0	0
REQUISITO 8	0	28.57142857	42.85714286	14.28571429	14.28571429	0

Classificação Geral

REQUISITO 8

Na tabela de somatório das respostas dos usuários, um exemplo da fórmula utilizada para seu cálculo foi: =SE(B2="M";1;0) + SE(C2="M";1;0) + SE(D2="M";1;0) + SE(E2="M";1;0) + SE(F2="M";1;0) + SE(G2="M";1;0) + SE(H2="M";1;0).

Na tabela de porcentagens das respostas dos usuários, um exemplo da fórmula utilizada para seu cálculo foi: =(C16/7)*100.

Após esses passos, apenas a tabela da direita é utilizada para geração dos gráficos pizza e em colunas através de funcionalidades do Excel.