Aplicação de uma técnica de visualização de dados baseado em árvores para auxiliar a priorização de requisitos em projetos ágeis

Fábio Abrantes Diniz[1], Thiago Reis da Silva[1], Íthalo Bruno Grigório de Moura[1], Diego Grosmann[1], Francisco Milton Mendes Neto[1], Pedro Fernandes Ribeiro Neto[1]

[1]Programa de Pós-Graduação em Ciência da Computação – PPgCC
Universidade do Estado do Rio Grande do Norte – UERN
Universidade Federal Rural do Semiárido – UFERSA
BR 110 – Km 46 – Bairro Costa e Silva – Campus Central
59.625-620 Mossoró – RN, Brasil

{fabio.abrantes.diniz, trsilva.si, ithalobgm, diegogrosmann}@gmail.com, miltonmendes@ufersa.edu.br, pedrofernandes@uern.br

Abstract. *The practice of prioritizing requirements involves analysis of the value of each requirement on the part of clients and selection of requirements that will be implemented in a particular version of the system. Wrong choices during the prioritization of requirements can affect the quality of the system, and consequently its acceptance by customers. Inserted in this context, this paper proposes a technique for data visualization, map-based tree (TreeMaps), to assist in visualizing the results of a practice that uses the technique of Kano to prioritize design requirements that address the agile Scrum methodology. For the results of this practice are difficult to be analyzed to determine the real importance of each requirement.*

Resumo. *A prática da priorização de requisitos envolve a análise da valorização de cada requisito por parte dos clientes e seleção dos requisitos que irão ser implementados em determinada versão do sistema. Escolhas erradas durante a priorização dos requisitos pode afetar a qualidade do sistema, e consequentemente sua aceitação pelos clientes. Inserido neste contexto, este artigo propõe uma técnica de visualização de dados, baseado em mapa de árvores (Treemaps), para auxiliar na visualização dos resultados de uma prática que utiliza a técnica de Kano para priorizar requisitos de projetos ágeis que abordam a metodologia Scrum. Pois, os resultados dessa prática são difíceis de serem analisados para determinar a real importância de cada requisito.*

1. Introdução

A priorização de requisitos é uma atividade desafiadora do desenvolvimento de software [Lamsweerde 2000]. Esta prática envolve a análise de cada requisito por parte dos *Stakeholders* (partes interessadas) e seleção dos requisitos que irão ser desenvolvidos em determinada versão do sistema. Nas metodologias ágeis, algumas práticas de priorização de requisitos têm sido adotadas em projetos ágeis para melhorar a etapa de

priorização dos requisitos. Entre elas, encontra-se uma prática de priorização de requisitos baseada na técnica de *Kano*, que foi aplicada em desenvolvimento de software que aborda a metodologia *Scrum* [Asfora 2009].

A técnica de *Kano* possibilita aos desenvolvedores de produtos transformarem as informações obtidas pelas pesquisas em melhorias reais no produto de forma a buscar a satisfação do cliente [Kano 1984]. No entanto, a técnica de *Kano* gera muita informação e a apresentação textual da mesma, por meio de tabelas e relatórios, faz com que os responsáveis gastem muito tempo processando grandes volumes de dados.

Portanto, este artigo utilizou uma técnica de visualização de dados baseada em Mapa de Árvore, conhecido como "*TreeMap*" [Jonhnson e Shneiderman 1991], com o objetivo de mostrar novas formas de visualização de dados, mais intuitivas e eficazes, para auxiliar a análise dos resultados originados da prática de priorização de requisitos baseada na técnica de *Kano*.

Este artigo encontra-se organizado da seguinte forma: na Seção 2 apresenta uma breve descrição da metodologia *Scrum*, detalhando as suas características usadas no gerenciamento dos projetos de software. Na Seção 3 descreve a prática adotada para a priorização de requisitos baseada na técnica de *Kano* e a sua adaptação para ambientes ágeis que utiliza a metodologia *Scrum*. Na Seção 4 aborda a técnica de visualização utilizada neste artigo, conhecida como *TreeMap*, para auxiliar a prática de priorização de requisitos em projetos ágeis, dando suporte na visualização dos dados gerados. Na Seção 5 é apresentado um protótipo chamado *TreeSolutions* para testar a técnica *TreeMap* e os seus resultados. As conclusões finais e trabalhos futuros são apresentados na Seção 6.

2. Scrum – Uma Metodologia de Desenvolvimento Ágil de Software

A metodologia ágil *Scrum* surgiu com a proposta de "desburocratizar" o processo de desenvolvimento de software, permitindo que as equipes sejam mais adaptáveis, respondendo rapidamente às constantes mudanças nos projetos de software. De acordo com seus evangelizadores, o cliente fica mais satisfeito, pois constantemente há entrega de funcionalidades desenvolvidas, e ele participa ativamente no projeto, trazendo seu conhecimento sobre o próprio negócio.

O *Scrum* se destaca dos outros processos ágeis por ser um método iterativo, incremental e ágil para o gerenciamento de Projetos [Schwaber 2004]. O processo inicia com os requisitos do projeto organizados em uma lista de requisitos, chamada de *Product Backlog,* em ordem decrescente de prioridade. A equipe separa uma parte do topo do *Backlog* para o *Sprint*, formando o *Sprint Backlog* (lista de tarefas do *Sprint*). O *Scrum* trabalha com desenvolvimento incremental, onde cada iteração é chamada de *Sprint*. Os *Sprints* são curtos, tendo duração de 30 dias.

Segundo [Schwaber 2004], durante um *Sprint,* a equipe tem autonomia para decidir como as tarefas serão implementadas e garante que os requisitos mais importantes sejam desenvolvidos primeiro. Além disso, a equipe tem curtas reuniões diárias, sempre no mesmo horário, junto com o *Scrum Master* (responsável pela gestão do projeto e liderança do grupo), chamadas de *Scrums*. Nessas reuniões é discutido o

andamento do trabalho, onde cada membro da equipe responde às questões: o que fiz desde ontem? O que pretendo fazer até amanhã?

A saída do *Sprint* é um conjunto de funcionalidades 100% desenvolvidas, que serão aprovadas pelo *Product Owner* (responsável pela definição do projeto, priorização dos requisitos e definição dos mesmos) e entregues ao cliente. Ao final de cada iteração, toda a equipe participa de uma retrospectiva do *Sprint.* Após a conclusão do *Sprint*, reinicia-se o ciclo, retirando-se a próxima fatia do *Product Backlog* para o próximo *Sprint* [Lee e Newcomb 1997].

3. A Técnica de *Kano* na Priorização de Requisitos para Projetos Ágeis

A aplicação da técnica de *Kano* no processo de priorização para projetos ágeis baseadas na metodologia *Scrum* é realizada antes do *Product Owner* enviar o *Product Backlog* para a equipe de desenvolvimento, a fim de que os requisitos primeiro possam ser priorizados. A técnica consiste em fazer um par de perguntas para cada requisito ao cliente. Uma pergunta de inclusão de um específico requisito e outra de exclusão deste específico requisito. Segundo [Asfora 2009], a importância da técnica de *Kano* em aplicar essas perguntas é mostrar o paralelo realizado entre o que deixa o cliente muito satisfeito e o que o deixa muito insatisfeito para sem ter a real noção de necessidade de cada requisito.

As perguntas são formadas da seguinte forma: (a) Como você se sentiria caso o Requisito X estivesse no próximo *release*? e (b) Como você se sentiria caso o Requisito X NÃO estivesse no próximo *release*?. As opções de respostas estão especificadas na Tabela 1.

Tabela 1: Respostas às perguntas de inclusão e exclusão do requisito e resultados para cada combinação [Asfora 2009].

		Perguntas de Exclusão do Requisito				
Perguntas de inclusão do requisito		Muito Satisfeito	Satisfeito	Pouco Satisfeito	Indiferente	Insatisfeito
	Muito Satisfeito	Q	D	D	D	L
	Satisfeito	R	I	I	I	M
	Pouco Satisfeito	R	I	I	I	M
	Indiferente	R	I	I	I	M
	Insatisfeito	R	R	R	R	Q

A categoria I (Indiferente) na Tabela 1 é utilizada quando o usuário demonstra que não tem necessidade real para esta funcionalidade, ou seja, tanto faz se ela é satisfeita ou não. A categoria M (Mandatório ou Indispensável) significa que se o requisito não forem feitos o cliente fica muito insatisfeito. Para o cliente, esses requisitos já estão embutidos nos produtos oferecidos, sendo, portanto, um pré-requisito. O fato de colocar os requisitos Mandatórios não tornará o cliente mais satisfeito, no entanto, sem eles o sistema não funciona e o cliente não adquire o produto. A categoria D (Desejado) significa que esses requisitos são requisitos que proporcionam grande satisfação ao cliente quando estão presentes, porém não representam insatisfação caso

não estejam presentes. A categoria L (Linear ou Importante) significa que a satisfação do cliente é proporcional ao nível de preenchimento desses requisitos, ou seja, quanto maior o nível de preenchimento, maior será a satisfação do cliente e vice-versa. A categoria R (Reverso) significa que, se aquele requisito for desenvolvido, poderá trazer uma rejeição ao software ou a determinada funcionalidade. A categoria Q (Questionável) significa que o usuário não entendeu as perguntas ou que ele não está correspondendo com a verdade [Asfora 2009]. De acordo com a análise de [Asfora 2009] mostra que a ordem de priorização dos requisitos que trariam um valor muito grande se forem atendidos é respectivamente: Mandatórios, depois os Importantes ou Lineares e, por fim, os Desejáveis.

De acordo com a Tabela 2, a combinação das respostas da tabela de *Kano* para as duas perguntas gera um resultado para cada requisito. Podemos observar que o requisito 1 (Req 1) possui a maior percentagem (43,8%) para classificação como Linear ou Importante. O requisito 3 (Req 3) foi classificado como Mandatório e Desejado, pois ambos possuem resultados próximos. Logo, deixando a seguinte ordem decrescente de priorização: Requisito 2, Requisito 1 e finalmente Requisito 3.

Tabela 2: Sumarização dos resultados.

Tema	D	L	M	I	R	Q	Classificação
Req 1	18,4	**43,8**	22,8	12,8	1,7	0,5	**Linear**
Req 2	8,3	30,9	**54,3**	4,2	1,4	0,9	**Mandatório**
Req 3	**39,1**	14,8	36,6	8,2	0,2	0,1	**Desejado Mandatório**

4. *TreeMap* – Uma Técnica de Visualização de Dados baseada em Árvore

TreeMap é uma técnica de visualização que explora conceitos básicos de ergonomia, fazendo com que o ser humano foque inicialmente seu olhar em figuras grandes para depois olhar para figuras pequenas [Jonhnson e Shneiderman 1991]. Essa característica pode ser utilizada na visualização dos itens do *Product Backlog*, com o objetivo de exibir a hierarquia dos requisitos que têm mais prioridade de serem desenvolvidos.

Algumas vantagens são evidentes na utilização *TreeMap*, tais como: utiliza eficientemente toda a tela de visualização; preserva o contexto geral da informação; navegação rápida entre os nós; permite ao usuário a visão geral do escopo e é muito útil na exibição de variáveis quantitativas dos dados [Jonhnson e Shneiderman 1991].

O *TreeMap* divide a tela de exibição em uma sequência aninhada de retângulos correspondentes a atributos de um conjunto de dados. Cada retângulo possui área e cor as quais são definidas por valores previamente estipulados. Logo possui propriedades que devem ser levadas em consideração e que, de acordo com [Jonhnson e Shneiderman 1991], são:

1. Um peso (classificação do requisito) determinará o tamanho de cada retângulo na estrutura;
2. A cor representará a satisfação do usuário em ter o requisito no sistema [Kano 1984]. As cores mais claras representarão os requisitos mais desejados pelos usuários, enquanto as cores mais escuras representarão os requisitos menos

desejados. Este artigo implementa uma escala de cores que atribui valores únicos para cada cor que os diferenciam;

3. Uso de *pop-ups* exibe informações do requisito quando o *mouse* é passado sobre a região do retângulo de interesse.

Outro ponto forte na construção do *TreeMap* é a definição do algoritmo a ser utilizado na criação dos retângulos. Os *layouts* dos retângulos dependem do algoritmo de divisão utilizado. Alguns algoritmos utilizados no *TreeSoltuions* foram: *Slice and Dice, Squarified TreeMap* e *Ordered TreeMap* [Jonhnson e Shneiderman 1991]. Estes algoritmos representam uma estrutura de dados hierárquica por divisão recursiva de retângulos. Essa recursão determina o *layout*, calculando a área do retângulo e preenche os retângulos em suas respectivas localidades.

5. TreeSolutions

Com base nos conceitos dos *TreeMap*, o protótipo *TreeSolutions* foi projetado para atender o *Product Owner* na etapa de priorização de requisitos da metodologia *Scrum* que utiliza a técnica de *Kano*. Ajudando o *Product Owner* nas tomadas de decisões na determinação do valor de negócio dos requisitos do *Product Backlog* a fim de que possa ser entregue à equipe de desenvolvedores mais rapidamente.

O protótipo é *open source* e desenvolvido na linguagem *Java*. Foram feitos testes com os dados presentes nos estudos de caso encontrados na dissertação elaborado por [Asfora 2009], e de acordo com a análise dos resultados observou-se que o uso da técnica *TreeMap* tornou a visualização dos dados mais eficiente. Nas subseções seguintes, é descrito o protótipo *TreeSolutions* com suas funcionalidades e o resultado obtido pelo *TreeMap* com os dados do estudo de caso da dissertação de [Asfora 2009].

5.1. Implementação do Protótipo TreeSolutions

O *TreeSolutions* consiste em um protótipo *desktop* desenvolvido na linguagem de programação *Java* para testar a técnica *TreeMap* nos dados vindos da tabela de *Kano*. Este protótipo está subdividido em dois módulos principais de implementação: Módulo *TreeMap* e Módulo Interface.

No Módulo Interface, o *Product Owner* tem acesso a uma interface gráfica com informações sobre seus *Product Backlog* contendo os requisitos. Esta interface possibilita inserção do *Product Backlog* (lista de requisito). Além disso, são inseridas as informações vindas dos resultados da tabela de *Kano*, tais como: os valores percentuais das categorias de todos os requisitos (Mandatório, Linear, Desejado, Questionável, Reverso ou Indiferente). As mesmas podem ser importadas e exportadas em arquivo no formato XML (*eXtensible Markup Language*).

No Módulo *TreeMap*, é implementada a técnica de visualização chamada *TreeMap*. Utilizou-se a classe *TreeMap* de *Java* que oferece funcionalidades adicionais de associar uma ordem aos elementos da coleção. Os algoritmos, citados na Seção 4, foram desenvolvidos e aplicados nesse módulo para representar as áreas dos retângulos e foram implementados os provedores de cores, que importam os modelos de cores RGB (*Red,Green e Blue*), HSB (*Hue, Saturation e Brightness*) e suas variações, para a reprodução das cores nas áreas dos retângulos do *TreeMap.*

De acordo com o contexto da implementação do protótipo, foi dada a área do retângulo como um atributo mais significativo que a cor, pois a área representa requisitos indispensáveis e importantes, ou seja, são requisitos que, sem eles, o sistema não funcionaria. A cor do retângulo representa a satisfação do usuário em ter o requisito no sistema, ou seja, quanto mais clara for a cor na escala das cores, maior é a satisfação do usuário em ter o requisito no sistema. A área do retângulo indica o grau de presença, isto é, retângulos de maior área representam os requisitos que têm que estar presentes no sistema. Logo, quanto maior for a área e maior for a cor na escala das cores, indica que esse requisito tem maior prioridade e deve ser implementado primeiro.

Para calcular o tamanho da área do retângulo, devido à sua função de servir para "medir o grau de presença no protótipo", foi feita uma média ponderada com as entradas dos dados da categoria Mandatório e Linear, com seus respectivos pesos dados como entrada ou por um padrão *default* do sistema escolhido. Os valores *default* são: Mandatório = 10, Linear = 5, desejado = 2, indiferente = 0, questionável = -1, reverso = -2. Estes valores foram atribuídos de acordo com os contextos de suas características e testes analisados durante o desenvolvimento. A categoria Questionável e o Reverso receberam valores negativos, pois categorizam requisitos que não trazem satisfação aos usuários [Asfora 2009].

Além de apresentar ao usuário a visualização dos requisitos em forma de retângulo, o protótipo possui o recurso de *pop-up,* que mostra informações do requisito quando o cursor do *mouse* passa sobre a área que representa um requisito, tais como: o tamanho da área do retângulo, o valor da cor, e os atributos do requisito com seus valores recebidos do resultado da tabela de *Kano*.

5.2. Resultados do Protótipo

O teste do protótipo *TreeSolutions* foi feito em cima dos dados coletados da dissertação de [Asfora 2009]. Os dados adquiridos envolvem um estudo de caso do CESAR (Centro de Estudos e Sistemas Avançados do Recife). Eles foram originados de um projeto do SEPG – Grupo de melhoria do processo interno de desenvolvimento de software. Os dados apresentados na Tabela 3 contêm 15 requisitos com seus resultados vindos da aplicação da técnica de *Kano*.

Tabela 3: Percentual das respostas advindas da técnica de *Kano*. Fonte: [Asfora 2009].

Requisito	Mandatório	Linear	Desejado	Indiferente	Reverso	Questionável
1	0	20	20	40	20	0
2	0	80	0	20	0	0
3	0	60	20	20	0	0
4	0	40	0	60	0	0
5	0	60	20	20	0	0
6	0	20	20	60	0	0
7	0	0	0	100	0	0
8	20	60	0	20	0	0
9	0	0	0	75	25	0
10	40	40	0	20	0	0
11	40	40	0	20	0	0
12	0	20	0	80	0	0
13	20	20	20	40	0	0
14	0	40	40	20	0	0

15	20	40	0	40	0	0

Portanto, a Figura 1 a seguir, ilustra o *TreeMap* resultante da entrada dos dados da Tabela 3. De acordo com a análise dos resultados, observou-se que o uso da técnica *TreeMap* tornou a visualização dos dados mais eficiente e robusta para a elaboração da lista de requisitos priorizados.

Como ilustra a Figura 1, a tela de visualização tem um campo de escolha da estratégia de visualização (a) em que se escolhe o algoritmo usado na construção do *TreeMap* e tem-se a escala de cores dos retângulos que podem ser alteradas conforme o gosto do usuário (d, e). A parte central da visualização (c) ilustra a lista de requisitos do projeto selecionado (f). Por último, quando se posiciona o cursor do *mouse* sobre um dos retângulos, aparece um *pop-up* mostrando informações do requisito (b).

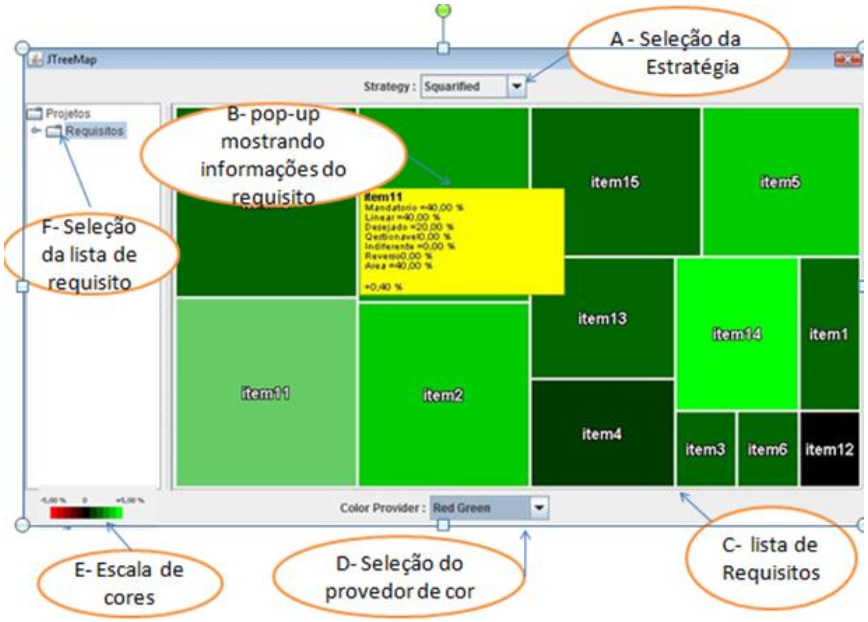

Figura 1: Visualização dos dados da Tabela 3 no *TreeMap*.

A Tabela 4 apresenta a consolidação dos resultados anteriores vindo da análise do *TreeMap,* mostrando tanto a lista de prioridade dos requisitos que devem ser selecionados para serem implementados pelos desenvolvedores quanto os requisitos que devem ficar em espera para futuras versões do software. Os requisitos mais a priori são os que têm menores valores na coluna Prioridade da Tabela 4. É importante lembrar que essa priorização foi formada única e exclusivamente com base na satisfação dos usuários que responderam a pesquisa. É prudente levar em conta dados como esforço e custo para implementar os requisitos antes de gerar a priorização final.

Tabela 4: Tabela de sugestão de priorização.

Prioridades	Selecionados	Em espera
1	Requisito 11	Requisito 1
2	Requisito 10	Requisito 4
3	Requisito 2	Requisito 6
4	Requisito 3	Requisito 7
5	Requisito 5	Requisito 9
6	Requisito 8	Requisito 12

7	Requisito 14	
8	Requisito 13	
9	Requisito 15	

6. Conclusões e Trabalhos Futuros

O objetivo principal deste artigo foi propor uma técnica de visualização utilizando *TreeMap* para ajudar uma prática de priorização de requisitos em projetos ágeis baseado na técnica de *Kano*. Como contribuição também foi desenvolvido um *Applet*, denominado de *TreeSolutions* para aplicação da técnica proposta. Comparado ao processo de visualização dos resultados de forma textuais, percebe-se claramente a melhoria que esta ferramenta traz ao usuário no que diz respeito à tomada de decisão.

O protótipo desenvolvido mostrou-se eficiente em realizar um *feedback* rápido na análise dos dados, gerando menos documentação e menos esforço na priorização dos requisitos. Além de ser simples de usar e fácil de entender. Logo, o *TreeSolutions* auxilia o *Product Owner* a priorizar seu *Product Backlog* mais eficiente e robusta. Consequentemente, as equipes de desenvolvedores podem estimar e desenvolverem os requisitos com mais facilidade.

Como trabalhos futuros pretende-se adicionar o protótipo a tabela de *Kano,* para que possa trazer as entradas de forma mais rápida para o *TreeMap*, gerando as visualizações das informações diretamente da tabela. Pretende-se também implementar outras técnicas de visualização de dados.

Agradecimentos

Os autores agradecem a CAPES pela concessão das bolsas de pesquisa e pelo apoio financeiro para realização da mesma.

Referências Bibliográficas

Asfora, D. M. (2009). “Uma abordagem para a priorização de requisitos em ambientes ágeis”. Dissertação de Mestrado, UFPE.

Johnson, B.; Shneiderman, B. (1991) “*Tree-maps: A spacelling approach to the visualization of hierarchical information structures*”. International IEEE Visualization Conference, v. 1, p. 284-291.

Kano, N. at al. (1984) “*Attractive Quality and Must-be Quality*”. Journal of the Japanese Society for Quality Control. v. 14, n. 14, p. 39-48.

Lamsweerde, A. (2000) “*Requirements Engineering in the year 2000: A Research Perspective*”. 22nd Proceedings of International Conference on Software Engineering. Limerick, Ireland.

Lee, M. C.; Newcomb, J. F. (1997) “*Applying the Kano methodology to meet customer requirements: NASA’s microgravity science program*”. Quality Management Journal, v. 4, n. 3, p. 95-110.

Schwaber, K. (2004) “*Agile Project Management with Scrum*”. Ed. Microsoft Press.