Priorização de requisitos e avaliação da qualidade de *software* segundo a percepção dos usuários

Aline Gomes Cordeiro
Mestre em engenharia de produção pela Universidade Estadual do Norte Fluminense Darcy Ribeiro. Professora do Instituto Federal Fluminense Darcy Ribeiro Centro de Ciências e Tecnologias, Laboratório de Engenharia de Produção. Campos dos Goytacazes, RJ – Brasil
E-mail: acordeiro@iff.edu.br

André Luís Policani Freitas
Doutor em engenharia de produção pela Universidade Estadual do Norte Fluminense Darcy Ribeiro, Brasil – Professor da Universidade Estadual do Norte Fluminense Darcy Ribeiro, Centro de Ciências e Tecnologias, Laboratório de Engenharia de Produção. Campos dos Goytacazes, RJ - Brasil
E-mail: andrepolicani@yahoo.com

Resumo

Atualmente *softwares* têm sido reconhecidos como importante ferramenta de apoio às diversas atividades e à tomada de decisões. No entanto, existem relatos a respeito de projetos de desenvolvimento de *software* fracassados. A questão problema apresentada por este artigo é a seguinte: Como é possível realizar a avaliação da qualidade de um produto de *software* desde as etapas iniciais do projeto, de forma que seja possível realizar as melhorias com menor esforço? O artigo traz uma abordagem metodológica para a priorização dos requisitos de *software* e a avaliação da qualidade do produto de *software*, segundo a percepção dos usuários. Em especial, a abordagem propõe o emprego da Análise Importância-Desempenho (IPA) e do método dos 100 pontos para a etapa de priorização, e para a etapa de avaliação de desempenho, o emprego da IPA e da escala contínua. Por meio de estudo de caso, a abordagem proposta foi aplicada a um projeto de desenvolvimento de *software* para gestão de recursos humanos. A partir desse uso foi possível captar os julgamentos, determinar as prioridades dos requisitos conforme a percepção dos usuários e sugerir ações relevantes com o objetivo de melhorar a qualidade do *software*. Acredita-se que a abordagem proposta seja aplicável ao desenvolvimento de produtos de *software* de pequeno porte.

Palavras-chave

Priorização de requisitos. Qualidade de software. Produto de software.

Prioritizaion of requirements and evaluation of software quality according to the users' perspective

Abstract

Nowadays software has been recognized as an important tool to support various activities and decision making. However, there are several reports regarding failures on development of software projects. In this scenario, the phase of requirements elicitation, allied to users' satisfaction, has been highlighted as leading to the improvement of the software development process. Thus, the main issue presented by this article is defined as follows: How is it possible to assess the quality of software product from the initial stages of design, so we can make improvements with less effort? This article proposes a methodological approach for prioritizing software requirements and evaluating the quality of software products according to the users' perspective. In particular, such approach proposes the use of Importance-Performance Analysis (IPA) and the 100-points method for prioritizing the requirements, and the use of IPA and the continuous scale for doing the performance evaluation stage. By conducting a case study, the proposed approach was applied to a software development project for human resource management. By doing so, users' evaluations were collected, the software requirements were prioritized according to their perspectives and relevant actions were suggested in order to improve the quality of the software. It is believed that the proposed approach may be suitable to the development of small software products.

Keywords

Requirements prioritization. Software quality. Software product.

INTRODUÇÃO

Atualmente, *softwares* têm sido cada vez mais utilizados nas organizações como instrumento de apoio às diversas atividades e à tomada de decisões. É possível observar sua presença como ferramenta em diferentes ramos de negócios, como saúde, educação e indústrias. As melhorias obtidas a partir de sua utilização podem ser notadas e geralmente justificam os investimentos necessários.

Apesar da importância do *software* para as organizações, um dos grandes desafios refere-se ao fato de que, às vezes, o investimento feito em sistemas informatizados não fornece o retorno esperado, ou seja, os sistemas não se adequam à realidade das empresas onde são implantados. Os processos de negócio nesses casos não são condizentes com os processos definidos pelo sistema informatizado.

Para Karlsson e Ryan (1996), um dos maiores riscos enfrentados por organizações que desenvolvem *software* comercial está associado ao não atendimento das necessidades e expectativas dos usuários. Para esses autores, esse risco pode ocasionar danos na reputação, perda de pedidos e redução dos lucros da empresa.

Pesquisas que visam à identificação das causas para o problema citado apontam a fase de elicitação de requisitos, também conhecida como levantamento de requisitos, como básica para a melhoria do processo. Ela é uma das atividades que ocorre no início do desenvolvimento de *software*. Erros gerados nessa etapa, se não forem corrigidos, estendem-se até o final do processo, e após a verificação de cada erro, todas as fases anteriores precisam ser refeitas. Para Sadraei *et al.* (2007), os requisitos de *software* são determinantes críticos da sua qualidade.

Nesse sentido, a avaliação da qualidade de *software* tem sido identificada como uma possibilidade de minimização do problema. A qualidade, como tratada atualmente na literatura científica, preocupa-se com a qualidade do processo de desenvolvimento ou com o produto final gerado. O foco de estudo deste artigo está relacionado ao produto. Este tipo de abordagem busca analisar a qualidade do produto obtido no desenvolvimento de *software*.

A abordagem baseada no produto tem contribuído para elevar a qualidade de *softwares*. No entanto, ela possui algumas limitações. A principal está relacionada ao fato de a avaliação da qualidade ser realizada apenas quando o produto já tiver sido implementado. Neste caso, após a avaliação são identificadas correções e melhorias. Porém, nesta etapa do projeto, o esforço necessário para alterar características do *software* é grande.

Caracteriza-se assim o problema de pesquisa tratado por este artigo: como é possível realizar a avaliação da qualidade do produto de *software* desde as etapas iniciais, de modo que seja possível realizar as melhorias com menor esforço?

Além da introdução, o artigo apresenta breve referencial teórico relacionado à avaliação da qualidade de *software* e ao seu processo de desenvolvimento; a abordagem metodológica proposta para avaliar a qualidade de *software,* desde a etapa de elicitação de requisitos com sua priorização, e a avaliação de desempenho do produto de *software*; os resultados de um estudo realizado com o intuito de investigar o emprego da referida abordagem na elaboração de um *software* de pequeno porte; a análise dos resultados segundo a percepção da gerência de projetos; as considerações finais; e por último, os apêndices contendo os questionários utilizados.

REFERENCIAL TEÓRICO

Qualidade de *software* e avaliação

É comum encontrar definições de qualidade de *software* que remetem ao fato de ele ter que atender aos requisitos, ou seja, às necessidades dos usuários, que variam de usuário para usuário, o que torna ainda mais complexa a obtenção de uma definição.

Para Denning (1992), a satisfação dos usuários é uma referência para a qualidade. Os usuários não se preocupam com detalhes estruturais, mas sim com o quanto o *software* facilita o trabalho.

Pressman (2002) cita a necessidade de conformidade aos requisitos funcionais, aos padrões de desenvolvimento claramente documentados e às características implícitas esperadas de todo *software* profissionalmente desenvolvido. A ausência de conformidade com os requisitos é falta de qualidade.

Segundo Magalhães (2006), para que um *software* seja considerado de qualidade é preciso que esteja em conformidade com os seus requisitos, atenda aos requisitos e expectativas do cliente e seja bem aceito por seus usuários.

Os estudos supracitados evidenciam a importância do papel do usuário para obter qualidade. Por isso, neste artigo considera-se que a percepção dos usuários é o foco para avaliação do *software*. Eles podem ser compreendidos como as pessoas que fazem ou farão uso do *software*.

De maneira complementar, é necessário ressaltar a distinção entre a qualidade voltada para o produto e a qualidade voltada para o processo de desenvolvimento. A primeira tradicionalmente se refere à avaliação do *software* após seu desenvolvimento. Dentre os estudos desenvolvidos visando à avaliação da qualidade do produto destacam-se os realizados por Punter *et al.* (2004); Jung (2007); Boente e Moré (2009); Larsen (2009); Saini *et al.* (2011); e Garofalakis *et al.* (2011).

Já a qualidade de *software* focada na abordagem para o processo almeja a qualidade do produto final pelo alcance da qualidade nas etapas intermediárias do processo de desenvolvimento. Em geral, metodologias que visam à qualidade do processo inserem tarefas e métodos que devem ser realizados durante cada etapa. Um exemplo típico são os métodos de teste de *software*. Dentre os estudos usando essa abordagem, destacam-se os feitos por Rosqvist *et al.* (2003); Belgamo *et al.* (2005); Gousios *et al.* (2007); Janzen e Saiedian (2008); Bertrán (2009); e Srivastava *et al.* (2011).

O processo de desenvolvimento de software

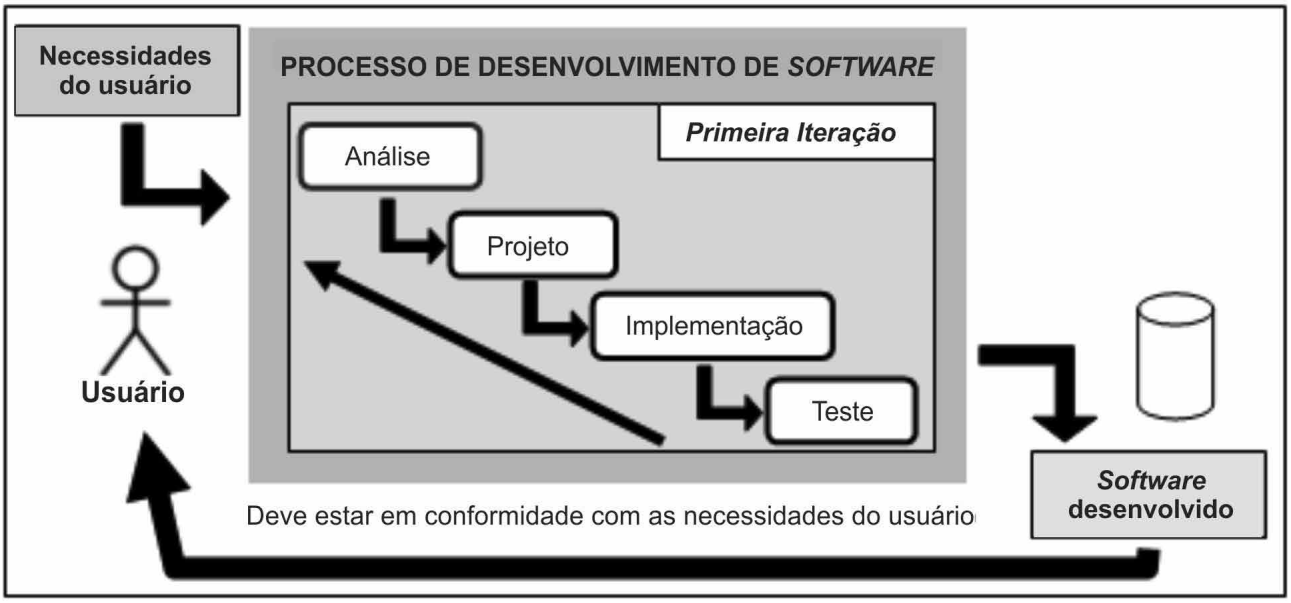
O aspecto não repetitivo do desenvolvimento de *software* torna essa atividade difícil e até mesmo imprevisível. Apenas pequena parcela da construção de *software* corresponde a atividades relativas à "montagem". Por isso, a atividade de desenvolvimento de *software* caracteriza-se como uma tarefa complexa (KOSCIANSKI e SOARES, 2007).

O processo é constituído por várias etapas e ações que devem ser realizadas com o objetivo de obter um produto de *software*. Ao longo do tempo, diversos modelos têm sido criados e testados. Segundo Pressman (2002), o modelo mais adequado para um projeto deve ser escolhido com base na natureza do projeto e da aplicação, nos métodos e ferramentas a serem usados, e nos produtos intermediários e finais requeridos.

O desenvolvimento iterativo incremental é uma das metodologias mais usadas para a implementação de *software*. Segundo esta abordagem, o desenvolvimento é constituído de uma série de miniprojetos, chamados de iterações. Cada uma visa à obtenção de um produto com qualidade superior oriundo da iteração anterior. Várias iterações ocorrem para que no final resulte um *software* pronto para ser utilizado (LARMAN, 2004). A figura 1, a seguir, ilustra esse processo.

Na figura 1 tem-se a representação das necessidades do usuário como ponto de partida para o desenvolvimento do *software*. A partir daí, o processo ocorre pela execução de suas etapas. No desenvolvimento iterativo, a cada iteração as etapas são executadas e uma entrega é feita aos usuários. Após várias iterações tem-se o produto pronto, mesmo sendo comum sofrer várias modificações para adaptação à realidade dos usuários ou para correção de erros.

FIGURA 1
Visão simplificada de um processo de desenvolvimento de software



Fonte: Cordeiro e Freitas (2008).

Metodologias que buscam a qualidade do produto de *software* definem atividades desempenhadas ao final do processo de desenvolvimento, com o *software* já testado. Já as metodologias que visam à qualidade do processo de desenvolvimento definem atividades realizadas durante o processo com a melhoria das etapas intermediárias: análise de requisitos; análise e projeto; implementação e teste. Tais métodos interferem no modo como os envolvidos executam as atividades.

A etapa de análise dos requisitos é um dos aspectos centrais para o entendimento do estudo descrito por este artigo. Por isso, as seções seguintes descrevem algumas atividades executadas nessa etapa.

Elicitação e priorização de requisitos

Um *software* deve possuir características que contribuam para a solução de problemas e a melhoria da qualidade de trabalho dos usuários, tendo como consequência a melhoria da qualidade do serviço ou do produto da empresa. Portanto, é preciso utilizar uma maneira adequada para identificar (elicitar) e priorizar tais aspectos, que constituem os requisitos.

Robertson e Robertson (2006) conceituam requisitos como algo que o produto deve fazer ou uma característica que o produto deve ter, e que é necessário ou desejado pelos *stakeholders*. Young (2004) corrobora esse conceito, afirmando que requisitos são atributos necessários em um sistema para que ele tenha valor e utilidade para os clientes e usuários. Para o autor, os requisitos do sistema são importantes porque fornecem a base para todo o trabalho de desenvolvimento de *software* subsequente.

Considerando que são as necessidades dos usuários que justificam o investimento em um projeto de *software*, não faz sentido que o foco principal do projeto seja outro senão a solução para essas demandas. Embora essa seja uma afirmativa lógica, a realidade mostra que é muito comum os objetivos de um projeto de *software* se distanciarem das necessidades de seus usuários. Por esta razão, a fase de elicitação de requisitos pode ser compreendida como base para todas as outras atividades relativas ao desenvolvimento de *software*.

Leffingwell e Widrig (1999) asseguram que erros na elicitação de requisitos representam a classe mais significativa de problemas relacionados ao desenvolvimento de *software*, sendo também os mais caros de se corrigir. Uma elicitação ineficaz traz consequências que podem levar ao fracasso do projeto. Isso pode ser explicado pelo fato de tal etapa constituir a base para as atividades subsequentes. Se a base é mal construída, as falhas decorrentes daí podem continuar acontecendo e até mesmo se tornar mais complexas posteriormente. Segundo Larman (2004), a indústria de *software* está repleta de projetos fracassados que não forneceram o que as pessoas realmente demandavam.

Segundo Boehm (1983), quando os problemas são detectados em fases mais avançadas, a correção se torna mais difícil e várias etapas precisam ser refeitas. O retrabalho tem consequências negativas para o projeto, como o aumento dos custos e atrasos no cronograma. Devido a esses aspectos, para Babar *et al.* (2011), a identificação correta dos requisitos interfere na qualidade do *software*.

Técnicas para a elicitação de requisitos são importantes porque facilitam a comunicação entre as pessoas envolvidas e devem ser definidas de acordo com as características dos requisitos e do negócio onde o *software* está inserido (COUGHLAN E MACREDIE, 2002). O quadro 1 apresenta técnicas descritas por Maiden e Rugg (1996).

As técnicas de elicitação visam à identificação dos requisitos. No entanto, devido a restrições de tempo e orçamento, pode ser difícil atender a todos os requisitos identificados para um sistema. Os requisitos geralmente são desenvolvidos em etapas e a priorização ajuda a definir quais devem ser implementadas primeiro (ALLEN *et al.*, 2008).

Segundo Karlsson, Wohlin e Regnell (1998), os requisitos devem ser alocados em diferentes versões do *software* e, para Berander (2004), a seleção "correta" dos requisitos que farão parte de cada versão é a etapa principal em direção ao sucesso de um projeto ou produto. Por isso, é preciso distinguir aqueles que terão maior impacto para a satisfação dos usuários. Alguns métodos destacados por Allen *et al.* (2008) são descritos no quadro 2.

Outro método utilizado para priorização de requisitos de *software* é a Análise Importância-Desempenho (*Importance-Performance Analysis* – IPA), proposta por Martilla e James (1977). Segundo Hudson, Hudson e Miller (2004), a IPA mostra a relevância de vários atributos e o desempenho do produto no que se refere ao estudo e análise desses atributos. Na aplicação do método, cada respondente deve julgar a importância de cada

QUADRO 1
Técnicas de elicitação de requisitos

Técnicas	Breve descrição
Entrevistas não estruturadas	Pergunta-se a um *stakeholder* a respeito do assunto. Não há uma lista de perguntas.
Entrevistas estruturadas	Prepara-se uma lista de perguntas para fazer ao *stakeholder*.
Brainstorming	*Stakeholders* geram quantas ideias forem possíveis, sem avaliar as ideias a priori.
Prototipagem rápida	*Stakeholders* comentam sobre um modelo em protótipo do sistema desejado.
Análise de cenário	Um cenário é uma descrição de uma sequência de ações e eventos para um caso específico de alguma tarefa genérica que o sistema deve realizar.

atributo, bem como a sua percepção de desempenho para cada um.

Leeworthy e Wiley (1996) explicam que a IPA oferece uma forma de visualização simples, na qual a apresentação dos dados é feita por meio de quatro quadrantes que formam a matriz da análise importância-desempenho. O eixo vertical da matriz é referente à importância e o horizontal à percepção de desempenho. Depois de realizada a mensuração, os atributos são dispostos na matriz e, de acordo com a posição onde ficam situados, obtém-se o resultado relativo à satisfação dos clientes.

Skok, Kophamel e Richardson (2001) realizaram a avaliação do sucesso de investimentos em sistemas de informação em uma organização da área de saúde utilizando a análise importância-desempenho. Segundo os autores, ela tem se revelado uma ferramenta de gestão simples e efetiva. O baixo custo de aplicação torna viável à sua aplicação em outras avaliações para acompanhamento das melhorias realizadas.

Cordeiro e Moll (2006) aplicaram a IPA para avaliar a qualidade de produto de *software* utilizando como critérios requisitos não funcionais. O método mostrou-se satisfatório para identificar os requisitos que necessitavam de melhorias.

A ABORDAGEM METODOLÓGICA PROPOSTA

A abordagem metodológica proposta sugere a avaliação da qualidade do produto de *software*, para que seja possível buscar a qualidade desde o início do desenvolvimento. Tal abordagem caracteriza-se por ser de uma pesquisa de natureza exploratória, tendo como objetivo prover percepções e compreensões a respeito de um problema. Segundo Malhotra (2006), a pesquisa exploratória é usada em casos nos quais é necessário definir o problema com maior precisão, identificar cursos relevantes de ação ou obter dados adicionais antes de poder criar uma abordagem conclusiva. A amostra, selecionada para gerar o máximo de discernimento, é pequena e não representativa.

QUADRO 2
Métodos para priorização de requisitos de *software*

Métodos	Breve descrição
Atribuição numérica	Utilizando uma escala de valores inteiros que varia de 1 a 5, os *stakeholders* devem identificar a qual nível da escala cada requisito corresponde.
Método dos 100 pontos	100 pontos devem ser d istribuídos entre os requisitos, atribuindo mais pontos aos mais importantes.
Triagem de requisitos	Processo que visa definir a prioridade relativa dos requisitos, estimar recursos para cada requisito e selecionar um subconjunto de requisitos para otimizar a probabilidade de sucesso do projeto.
AHP (*Analytic Hierarchy Process*)	Método de apoio à decisão utilizado em situações nas quais múltiplos objetivos estão presentes. Utiliza a comparação paritária para calcular o valor (importância) relativo de cada item (requisito).

Neste artigo, o caráter exploratório advém do fato de que não há um consenso entre pesquisadores e desenvolvedores acerca do processo mais adequado para a elaboração de produtos de *software*. Além disso, não é possível transpor os resultados obtidos neste estudo para outras situações, devido a eventuais diferenças entre as realidades existentes.

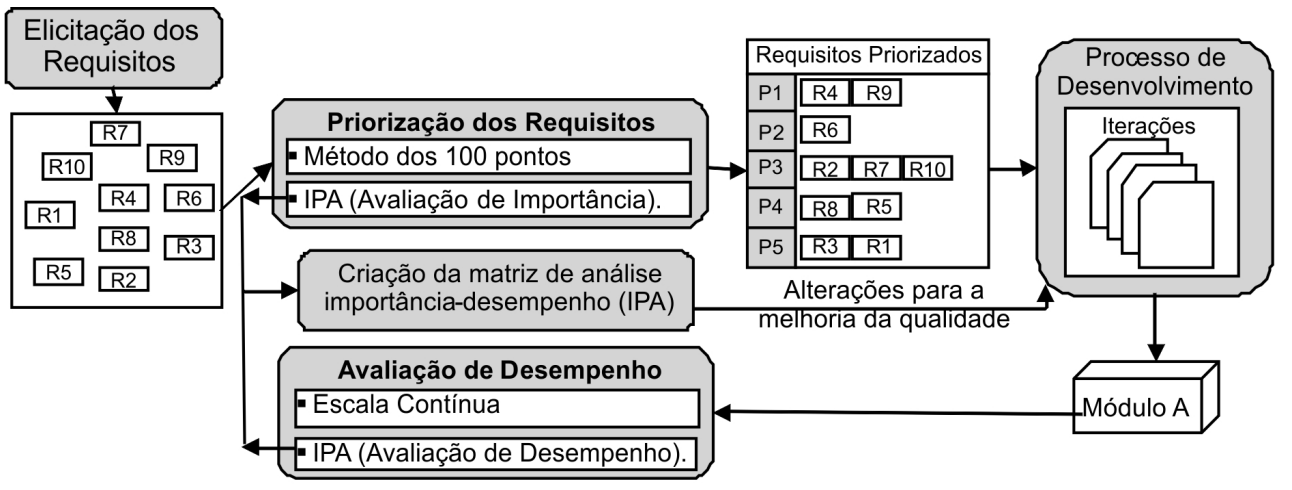
Apesar de existir a preocupação com a qualidade desde o início, não seria correto afirmar que a referida abordagem é baseada no processo, pois o foco do estudo não está nas atividades executadas em cada etapa do desenvolvimento. O foco concentra-se na satisfação do usuário em relação ao produto de *software* resultante do processo. A figura 2 representa o modelo conceitual que sintetiza a abordagem proposta e relaciona conceitos relevantes para o entendimento do estudo de caso.

Acredita-se que a conformidade do *software* aos requisitos seja essencial para a satisfação dos usuários e que a satisfação dos usuários, por sua vez, seja determinante para a melhoria da qualidade do *software*. Neste sentido, propõe-se a priorização dos requisitos no início do desenvolvimento e a avaliação de desempenho quando o produto já está em uso, conforme ilustrado na figura 2. Após a elicitação dos requisitos, obtém-se uma lista de requisitos que precisam ser desenvolvidos. Contudo, neste momento, os desenvolvedores não sabem quais requisitos são mais relevantes para os usuários, por isso é necessário priorizá-los. Para atender aos requisitos de acordo com a percepção dos usuários, utiliza-se o método dos 100 pontos em conjunto com a avaliação de importância da análise IPA.

A priorização é relevante porque os usuários têm expectativas em relação ao *software* que está sendo gerado. A entrega dos requisitos segundo a percepção dos usuários, além de contribuir para a satisfação deles, pode ser essencial para a continuidade do projeto. Para exemplificar, na figura 2 estão representados cinco níveis de prioridades, variando de P1 a P5.

Caso o *software* seja elaborado de forma iterativa, várias iterações podem ocorrer até que o primeiro módulo seja considerado "pronto". Nesta circunstância, entregas intermediárias devem ser feitas para que os usuários comecem a utilizá-lo. Quando todos os requisitos ou a maior parte deles já tiver sido desenvolvida e o módulo estiver em uso, é possível realizar a avaliação de desempenho do produto, utilizando-se uma escala contínua e a avaliação de desempenho da análise IPA.

FIGURA 2
Modelo Conceitual



Os resultados da avaliação de importância e da avaliação de desempenho da IPA devem ser usados para a construção da matriz de análise importância-desempenho. A matriz permite a identificação dos atributos, neste caso requisitos, que requerem melhorias. Alterações necessárias podem ser realizadas no módulo "A".

Após as atividades de melhoria, novas avaliações de desempenho podem ser feitas com o intuito de obter um produto mais adequado à percepção dos usuários. A seguir, são apresentadas as etapas da abordagem que constituíram o estudo de caso efetuado para avaliar a qualidade de *software* segundo a percepção dos usuários. Os resultados obtidos estão descritos em cada fase.

Identificação de uma equipe de desenvolvimento

Como este artigo busca a qualidade de *software* desde o início do desenvolvimento, foi preciso acompanhar o trabalho de uma equipe que estivesse começando um projeto. O estudo de caso foi realizado na criação de *software* de uma fundação situada no município de Campos dos Goytacazes, Estado do Rio de Janeiro. A fundação é mantenedora de unidades de saúde vinculadas ao Sistema Único de Saúde (SUS). O objetivo é elaborar um *software* que apoie as atividades de gestão de recursos humanos (RH).

Elicitação de requisitos

A escolha das técnicas para elicitação de requisitos depende da equipe e das características do projeto. Para o *software* em estudo foram utilizadas quatro das técnicas descritas anteriormente: i) entrevista não estruturada; ii) entrevista estruturada; iii) prototipagem rápida; iv) análise de cenário. Essas técnicas foram consideradas adequadas pela equipe de desenvolvimento, pois permitiram o entendimento do domínio onde o *software* se insere. Uma lista inicial contendo 12 requisitos foi identificada.

Priorização dos requisitos

De acordo com o contexto da aplicação, os requisitos foram atribuídos a um dos cinco grupos. Um questionário contendo os requisitos foi elaborado de acordo com os métodos definidos para priorização: a análise IPA (etapa de avaliação da importância) e o método dos 100 pontos (Ver Apêndice 1). Visando obter dados confiáveis e para evitar que os usuários interferissem nas opiniões alheias, os questionários foram aplicados individualmente a 10 pessoas que atuavam no domínio onde o *software* seria implantado.

a. Priorização dos requisitos segundo a IPA

Os usuários avaliaram a importância de cada requisito utilizando uma escala ordinal de 5 pontos, cujos conceitos e valores foram denotados por: nada importante (1), pouco importante (2), neutro (3), importante (4) e muito importante (5). Para cada requisito, a tabela 1, a seguir, apresenta a frequência de atribuição dos julgamentos às categorias da escala de avaliação de importância da IPA, a média da importância e a ordem de importância (prioridade). O requisito 1.1 (registro de frequência) foi considerado mais importante (ordem de prioridade 1). Já o requisito 5.2 (solicitação e entrega de declarações) foi considerado o menos importante. Além disso, nota-se uma tendência relacionada à aplicação da IPA: os avaliadores tendem a atribuir altos valores de importância para os itens. Esta tendência também foi notada por Ainin e Hisham (2008) e Duke (2002).

b. Priorização dos requisitos segundo o método dos 100 pontos

Cada usuário distribuiu 100 pontos, atribuindo pontuação maior aos requisitos considerados mais importantes. A tabela 2, a seguir, apresenta a média das pontuações para cada requisito. Assim como o resultado obtido por meio da IPA, o requisito 1.1 foi o de maior prioridade.

TABELA 1
Importância média dos requisitos segundo a IPA

	Requisitos											
Grau de importância	**1.1**	**1.2**	**1.3**	**2.1**	**2.2**	**3.1**	**3.2**	**3.3**	**4.1**	**4.2**	**5.1**	**5.2**
1- Nada Importante	0	0	0	0	0	0	0	0	0	0	0	0
2- Pouco Importante	0	0	0	0	0	0	0	0	0	0	0	0
3- Neutro	0	0	0	0	2	0	0	0	0	0	0	0
4- Importante	2	6	4	3	3	5	5	5	6	5	7	8
5- Muito Importante	8	4	6	7	5	5	5	5	4	5	3	2
Total	10	10	10	10	10	10	10	10	10	10	10	10
Média	4,8	4,4	4,6	4,7	4,3	4,5	4,5	4,5	4,4	4,5	4,3	4,2
Ordem	1	8	3	2	10	4	4	4	8	4	10	12

A utilização de dois métodos resultou em duas listas distintas de requisitos priorizados, conforme observado no quadro 3, a seguir. É possível afirmar que o requisito mais relevante está relacionado à necessidade de controle de frequência dos funcionários da instituição. Para os demais itens, houve variação em relação à prioridade. O resultado pode ser explicado pelo fato de alguns usuários terem julgado aspectos como "muito importante". No entanto, no método dos 100 pontos, eles forneceram julgamentos diferenciados. Vale ressaltar que o esforço cognitivo para o julgamento pelo método dos 100 pontos pode ser considerado maior do que o necessário para a escolha de uma das categorias da escala ordinal de importância. Neste caso, onde há divergência, recomenda-se a utilização do resultado do método dos 100 pontos.

Desenvolvimento

O desenvolvimento se deu tendo como referência a lista de requisitos. A evolução do processo foi acompanhada pelos usuários que validaram as funcionalidades, sugeriram melhorias e identificaram possíveis erros. Como a metodologia utilizada pelo projeto em questão é o desenvolvimento iterativo incremental, os usuários tiveram acesso aos requisitos identificados gradativamente e puderam iniciar o uso do *software* pelos aspectos considerados mais importantes.

Após o desenvolvimento do *software* ou após a realização de várias iterações, sugere-se que seja verificado por um *checklist* se os requisitos prioritários foram atendidos. Só após tal verificação deve-se implantar e avaliar o desempenho do *software*, visto que a ausência de requisitos prioritários pode causar insatisfação nos usuários, o que é prejudicial para o projeto como um todo.

Para a realização do *checklist,* utilizou-se o rol de requisitos definidos a partir do método dos 100 pontos. Apenas os itens 5.1 e 5.2 não foram observados, por uma decisão gerencial. Os gestores perceberam, em reuniões com os usuários, que as necessidades relativas a esses requisitos não estavam no contexto de aplicação do *software* (tratam de aspectos relativos a vários setores e não apenas aos setores ligados aos recursos humanos). Vale destacar que esses requisitos não foram considerados prioritários pelos usuários.

TABELA 2
Resultados da distribuição dos 100 pontos

	Requisitos											
	1.1	**1.2**	**1.3**	**2.1**	**2.2**	**3.1**	**3.2**	**3.3**	**4.1**	**4.2**	**5.1**	**5.2**
Soma	146,3	89,3	73,3	89,3	60,3	67,3	73,3	77,3	90,3	100,3	68,8	63,8
Média	14,6	8,9	7,3	8,9	6,0	6,7	7,3	7,7	9,0	10,0	6,9	6,4
Ordem	1	4	7	4	12	10	7	6	3	2	9	11

QUADRO 3
Lista de requisitos priorizados

Ordem	Avaliação de Importância (IPA)	Método dos 100 pontos
1	1.1 Frequência	1.1 Frequência
2	2.1 Informações pessoais	4.2 Controle de Afastamentos
3	1.3 Relatório de frequência	4.1 Controle de Férias
4	3.1 Crachás; 3.2 Informações sobre convocação; 3.3 Alocação de funcionários; 4.2 Controle de afastamentos	1.2 Escalas e horários de trabalho; 2.1 Informações pessoais
5	1.2 Escalas e horários de trabalho; 4.1 Controle de férias	3.3 Alocação de funcionários
6	2.2 Registro de deficiências; 5.1 Entrada e saída de carteiras de trabalho	1.3 Relatório de frequência; 3.2 Informações sobre convocação
7	5.2 Solicitação e entrega de declarações	5.1 Entrada e saída de carteiras de trabalho
8		3.1 Crachás
9		5.2 Solicitação e entrega de declarações
10		2.2 Registro de deficiências

Implantação

A primeira versão do *software* entregue aos usuários foi disponibilizada no dia 23 de outubro de 2008, mas eles começaram a utilizar o sistema no dia 12 de janeiro de 2009. O atraso está associado à limitação de funcionários no setor. Entre outubro de 2008 e outubro de 2009 ocorreram aproximadamente 150 atualizações no *software*. Destas, 64 foram externas (percebidas pelos usuários e, em alguns casos, derivadas de solicitações de usuários).

Avaliação de desempenho

Elaborou-se um questionário composto pela lista de requisitos desenvolvidos (Apêndice 2) e que permitiu o uso da escala contínua e da análise IPA (avaliação de desempenho). O questionário foi aplicado a 14 usuários em outubro de 2009, ou seja, um ano após o questionário de priorização.

A experiência dos usuários em relação ao *software* é uma característica que influencia a capacidade de avaliar o desempenho à luz dos requisitos. Neste sentido, foram considerados os registros das ações

de inserção, exclusão e alteração realizadas por usuário, a partir de determinado computador, além da data e hora de ocorrência.

A Análise dos Quartis, proposta por Freitas, Manhães e Cozendey (2006) foi utilizada para classificar os usuários em relação à quantidade de ações realizadas. A técnica usa a medida de posição denominada Quartil para atribuir itens em quatro níveis de prioridade de intervenção (crítica, alta, moderada ou baixa) e é empregada com sucesso em diversos estudos (Freitas, Rodrigues e Costa (2009) e Freitas, Bolsanello e Viana (2008), por exemplo), o que motivou sua utilização. Os Quartis são interpretados como valores de fronteira, ou seja, valores que separam cada nível de prioridade. Neste estudo, usuários que realizaram ações em menor quantidade do que o primeiro *Quartil* foram descartados (figura 3). Dos 14 questionários respondidos, 10 foram considerados.

a. Avaliação de desempenho segundo a IPA

Os usuários avaliaram o desempenho de cada requisito utilizando uma escala ordinal de 5 pontos cujos valores e conceitos foram denotados por muito ruim (1), ruim (2), neutro (3), bom (4) e muito bom (5). Para cada requisito, a tabela 3, a seguir, apresenta a frequência dos itens da escala de julgamentos da avaliação de desempenho da IPA, as médias de desempenho e a ordem de prioridade. O requisito 2.1 obteve a maior média de desempenho (ordem 1). Os requisitos 1.2, 3.2 e 4.2 tiveram as menores médias.

Para identificar os requisitos críticos por meio da IPA, as análises de importância e desempenho devem ser feitas em conjunto. Assim como no estudo realizado por Hudson *et al.* (2004), as avaliações de importância e desempenho da IPA foram feitas em momentos distintos. A tabela 4, a seguir, mostra as médias para cada requisito.

Segundo Magal e Levenburg (2005), a interseção dos eixos da matriz IPA pode ser definida a partir das médias globais de importância e desempenho. Aqui, essas médias foram, respectivamente, 4,52 e 3,74. A figura 4, a seguir, apresenta a matriz IPA, na qual cada requisito está representado por um símbolo de acordo com o quadrante onde se situa.

Os requisitos 1.1 e 2.1 estão posicionados no quadrante "manter o bom trabalho". São requisitos prioritários, mas apresentam bom desempenho segundo a percepção dos usuários. Nessas circunstâncias, ações de melhoria nesse conjunto não são prioritárias.

Os requisitos 2.2, 3.3 e 4.1 se situam no quadrante "possível excesso". São aspectos menos urgentes na avaliação de importância, mas que tiveram bom desempenho. A localização nesse quadrante indica que recursos podem estar sendo alocados aos itens de forma indevida, ou seja, os mesmos recursos poderiam ser usados para melhorar o desempenho de itens prioritários. No entanto, neste estudo, os requisitos já foram desenvolvidos e os recursos já foram empregados.

FIGURA 3
Classificação dos usuários segundo a quantidade de ações realizadas (*Análise dos Quartis*)

Classificação dos usuários segundo quantidade de ações realizadas													
Freqüência Baixa				Freqüência Moderada			Freqüência Alta			Freqüência Muito Alta			
U12	U14	U13	U15	U6	U4	U7	U8	U9	U3	U10	U1	U2	U11
1	3	9	28	33	162	392	450	466	921	939	1919	2774	4084
			1º Q = 29,25		2º Q = 421				3º Q = 934,5				

TABELA 3
Desempenho médio dos itens segundo a Análise Importância-Desempenho (IPA)

	Requisitos									
Grau de desempenho	**1.1**	**1.2**	**1.3**	**2.1**	**2.2**	**3.1**	**3.2**	**3.3**	**4.1**	**4.2**
1- Muito Ruim	0	1	1	0	0	0	0	0	0	1
2- Ruim	0	2	0	0	0	3	2	0	1	1
3- Neutro	1	2	3	1	4	2	3	1	2	4
4- Bom	6	5	4	5	4	2	4	6	3	2
5- Muito Bom	3	0	2	4	2	3	1	3	4	2
Total	10	10	10	10	10	10	10	10	10	10
Média	4,2	3,1	3,6	4,3	3,8	3,5	3,4	4,2	4,0	3,3
Ordem	2	10	6	1	5	7	8	2	4	9

TABELA 4
Médias de importância (I) e desempenho (D) para os requisitos

Requisitos																			
1.1		**1.2**		**1.3**		**2.1**		**2.2**		**3.1**		**3.2**		**3.3**		**4.1**		**4.2**	
D	**I**	**D**	**I**	**D**	**I**	**D**	**I**	**D**	**I**	**D**	**I**	**D**	**I**	**D**	**I**	**D**	**I**	**D**	**I**
4,2	4,8	3,1	4,4	3,6	4,6	4,3	4,7	3,8	4,3	3,5	4,5	3,4	4,5	4,2	4,5	4,0	4,4	3,3	4,5

Os requisitos 1.2, 3.1, 3.2 e 4.2 possuem baixa prioridade porque, comparados aos demais, possuem menor importância e baixo desempenho. Apenas o requisito 1.3 está posicionado no quadrante "concentrar aqui", onde as necessidades de melhoria são prioritárias. No entanto, é preciso notar que outros requisitos estão situados bem próximos a esse quadrante (requisitos 1.2, 3.1, 3.2 e 4.2). Sugere-se que esses requisitos sejam cuidadosamente observados, pois um pequeno aumento na importância também os deslocaria para o quadrante de ações prioritárias.

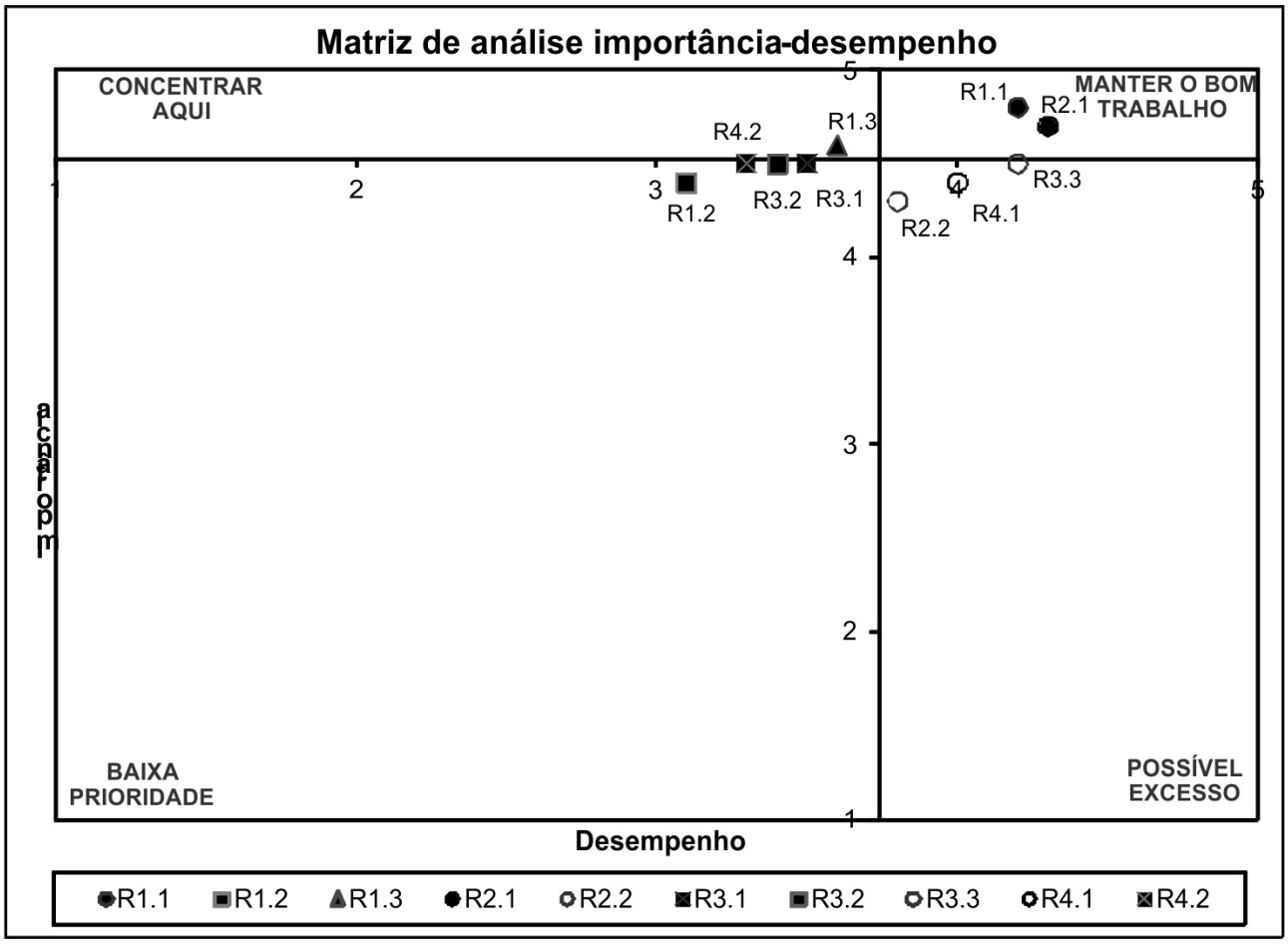
b. Avaliação de desempenho segundo a escala contínua

A escala contínua é representada por uma reta cujos valores possíveis variam de zero (muito ruim) a cem pontos (muito bom), onde os usuários marcaram um valor representativo da sua percepção acerca do desempenho do *software* à luz de cada requisito (Apêndice 2). A tabela 5, a seguir, traz o desempenho médio de cada requisito. Com os resultados obtidos pela matriz IPA, nota-se que os requisitos 1.2, 3.1, 3.2 e 4.2, situados próximos ao quadrante "concentrar aqui" também tiveram as piores médias pela escala contínua. Entretanto, o requisito 1.3, que segundo a matriz IPA necessita de ações urgentes, na escala contínua é o quinto em ordem de prioridade. Apesar disso, para planejamento das ações de melhoria, recomenda-se considerar críticos os cinco requisitos.

Definição e execução das melhorias para os requisitos críticos

A técnica 5W1H foi utilizada para identificar possíveis ações de melhoria para os requisitos

FIGURA 4
Matriz de Análise Importância-Desempenho



críticos. A exemplo do método utilizado por Dantas *et al.* (2005), utilizou-se um sistema de controle de versões para a coleta das informações necessárias para compor o 5W1H. O quadro 4 apresenta ações de melhorias referentes aos requisitos 1.2 e 3.1. A coluna "o que" representa possibilidades de melhoria ou correção de erros identificados. A coluna "por que" justifica a necessidade de implementação da correção. A coluna "quando" identifica a data de identificação da necessidade de alteração. A coluna "quem" indica quem identificou a ação a ser realizada - identificam-se

TABELA 5
Médias de desempenho obtidas pela escala contínua

	Requisitos									
	1.1	**1.2**	**1.3**	**2.1**	**2.2**	**3.1**	**3.2**	**3.3**	**4.1**	**4.2**
Soma	688	581	691	737	706	680	626	769	741	566
Média	68,8	58,1	69,1	73,7	70,6	68	62,6	76,9	74,1	56,6
Ordem	6	9	5	3	4	7	8	1	2	10

os desenvolvedores pela letra "D" e os usuários por "U". Finalmente, a coluna "como" descreve resumidamente como a alteração deve ser feita.

Para o requisito 1.2 foram percebidas apenas duas possibilidades de melhorias. As alterações necessárias no requisito 3.1 caracterizam-se por intervalos de tempo relativamente longos (dois meses) para identificar as ações. Esta pode ser considerada uma característica negativa porque revela que os usuários não utilizaram o requisito nesse período.

Nova avaliação de desempenho

No caso do *software* em questão, nova avaliação de desempenho pode ser feita após a execução de todas as ações de melhoria identificadas.

ANÁLISE SEGUNDO A GERÊNCIA DO PROJETO

Uma reunião foi realizada com o intuito de notar a opinião do gerente do projeto a respeito dos resultados obtidos. Destacam-se as seguintes considerações:

• **Discordância em relação aos resultados de dois requisitos:** Segundo o gerente, o requisito 3.1 (emissão de crachás) tem desempenho muito bom, atende às necessidades dos usuários, não apresenta falhas e, por conseguinte, o sistema agiliza o processo de emissão de crachás. No entanto, o processo de negócio relativo a esse requisito mostra algumas dificuldades. Os usuários "confundem" isso e associam as dificuldades ao sistema. Já o requisito 4.1 (controle de férias) é mais importante do que os

QUADRO 4
5W1H para os requisitos 1.2 e 3.1

	O que (*what*)?	Por que (*why*)?	Quando (*when*)?	Quem (*who*)?	Como (*how*)?
R 1.2	Implementar escala para os funcionários plantonistas.	A escala é o único jeito de saber quando um funcionário plantonista deve trabalhar.	08/09/09	D3	Implementação de uma função no sistema
	Colocar no sistema uma possibilidade de abonar a redução de carga horária.	Devido à inconsistência entre cargas horárias.	23/09/09	U16	Implementação de uma melhoria no sistema
R 3.1	Criar crachás distintos por unidade.	Para identificar de forma mais rápida de qual unidade é o funcionário.	30/01/09	U2	Implementação de uma melhoria no sistema
	Alterar crachá, mostrar local de trabalho.	Antes mostrava o setor, mas é importante que o crachá identifique onde o funcionário atua.	03/04/09	U1	Implementação de uma melhoria no sistema
	Crachá - aumentar altura do código de barras e área em branco.	Para facilitar a leitura pelo relógio de ponto.	30/06/09	U2	Correção de um erro no sistema
	Alterar crachá permitindo impressão mesmo quando o funcionário é de outra unidade.	Só estava sendo possível imprimir crachás de funcionários lotados na unidade principal.	16/10/09	U2	Correção de um erro no sistema
	Permitir a impressão de uma lista de crachás impressos.	Para facilitar o controle de impressão e entrega de crachás.	19/10/09	U2	Implementação de uma função no sistema

usuários avaliaram, visto que se trata de um processo bastante complexo e básico para o funcionamento da instituição. Além do mais, há o que melhorar nesse item. Logo, o desempenho do requisito seria pior do que o informado pelos usuários.

• **Dificuldade em seguir as prioridades:** A priorização dos requisitos é essencial para o desenvolvimento de *software* porque os desenvolvedores podem focar nos requisitos mais importantes. Porém, na instituição onde o estudo foi realizado existem usuários de diferentes níveis gerenciais. O diretor da fundação, por exemplo, pode determinar que um item seja visto primeiro, e solicitações desse tipo precisam ser atendidas. Por esse motivo, nem sempre é possível seguir exatamente a ordem de prioridades apontada pelos usuários.

• **Relação de dependência entre usuários e sistema:** A priorização dos requisitos de acordo com a percepção dos usuários permite que tenham contato com requisitos importantes desde o início, o que de certo modo possibilita que estabeleçam uma relação de dependência com o sistema. Essa característica pode ser considerada relevante para a continuidade do projeto de elaboração do *software*.

• **Dependência entre requisitos:** A dependência entre os requisitos pode ser ressaltada como um fator que dificulta a aplicação da abordagem metodológica. Ou seja, como desenvolver o requisito A prioritário, se para ele funcionar o requisito B deve estar pronto? Neste caso, os criadores tendem a desenvolver logo o requisito B. Para seguir as prioridades em situações desse tipo, é necessário atender basicamente os requisitos menos prioritários, inserindo posteriormente regras de negócio, por exemplo.

CONSIDERAÇÕES FINAIS

Um dos maiores problemas relatados na literatura a respeito dos projetos de desenvolvimento de *software* refere-se a sua não conformidade aos requisitos desejados pelos usuários. Considerando essa dificuldade, este artigo apresentou uma abordagem metodológica para avaliar a qualidade de produtos de *software* desde as fases iniciais, tendo sempre como foco a percepção dos usuários.

Acredita-se que a satisfação dos usuários seja preponderante para o sucesso de um projeto de e a continuidade de utilização de um *software*. Por isso, a abordagem buscou o alinhamento entre as necessidades dos usuários e o *software* criado. Espera-se que essa abordagem permita aumentar as chances de sucesso dos projetos de desenvolvimento e elevar a qualidade dos produtos de *software*.

O estudo objetivou verificar a viabilidade de aplicação da abordagem proposta. A partir dessa aplicação, conclui-se que é possível usá-la em projetos de desenvolvimento de *software* de pequeno porte. Os resultados sugerem a obtenção de dois benefícios:

a) priorização dos requisitos com base na percepção dos usuários do *software*, permitindo que os desenvolvedores tivessem uma base metodológica para conduzir o trabalho de desenvolvimento, tendo como referência os requisitos mais relevantes de acordo com a percepção dos usuários, e;

b) estabelecimento de um método para avaliar o desempenho do *software*, tendo como critérios os requisitos funcionais, importantes para percepção dos usuários em relação ao *software*. Em geral, os usuários criam expectativas relacionadas a esse tipo de requisito. Ações foram sugeridas visando à melhoria contínua do *software*.

Entretanto, algumas dificuldades e limitações foram encontradas. A principal delas refere-se ao fato de que nem sempre são os desenvolvedores que decidem a ordem de atendimento dos requisitos. Ou seja, influências externas podem impedir que os responsáveis sigam exatamente a lista de requisitos priorizada.

Por tratar-se de um estudo de caso, os resultados são específicos para o contexto onde a aplicação foi feita. Por exemplo, não é possível afirmar que em um projeto de grande porte os resultados serão satisfatórios. Além disso, as equipes participantes (desenvolvedores e usuários) foram reduzidas (amostras pequenas), o que restringe a realização de análises estatísticas mais elaboradas. Contudo, buscou-se envolver todos os usuários do setor cliente.

Em relação aos métodos utilizados, segundo Duke e Mount (1996), a matriz de análise importância-desempenho possui como limitação a falta de testes de significância estatística.

Por fim, uma das características do desenvolvimento de *software* é a capacidade de os requisitos mudarem, seja por fatores externos ou internos. Essas alterações podem dificultar a etapa de priorização dos requisitos, caracterizando-se como mais uma dificuldade enfrentada pela abordagem proposta e técnicas correlatas.

Agradecimentos

Os autores agradecem o apoio concedido pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

Data de submissão: 26-10-2010 **Data de aceite: 12-07-2012**

REFERÊNCIAS

AININ, S.; HISHAM, N.H. Applying Importance-Performance Analysis to Information Systems: An Exploratory Case Study. *Journal of Information, Information Technology, and Organizations,* v. 3, p. 95-103, 2008.

ALLEN, J.H.; BARNUN, S.J.; ELLISON, R.J.; MCGRAW, G.; MEAD, N.R.. *Software security engineering:* a guide for project managers. Upper Saddle River, NJ : Addison-Wesley. 2008. 368 p.

BABAR, M. I.; RARNZAN, M.; GHAYYUR, S. A. K. Challenges and Future Trends in Software Requirements Prioritization. In: COMPUTER NETWORKS AND INFORMATION TECHNOLOGY INTERNATIONAL CONFERENCE (ICCNIT'11), 2011. *Proceedings*... p. 319-324.

BELGAMO, A.; FABBRI, S.; MALDONADO, J. C. Avaliando a qualidade da técnica GUCCRA com técnica de inspeção. In: WORKSHOP ON REQUIREMENTS ENGINEERING, 7., 2005, Porto. *Proceedings*…

BERANDER, P. *Prioritization of Stakeholder Needs in Software Engineering Understanding and Evaluation*. Thesis (Licentiate of Technology in Software Engineering) - Department of Systems and Software Engineering, Blekinge Institute of Technology, Sweden, 2004, 172p.

BERTRÁN, I.M. *Avaliação da qualidade de software com base em modelos UML*. Dissertação (Mestrado) - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2009.

BOEHM, B.W. Seven basic principles of software engineering. *The Journal of Systems and Software.* v. 3. p. 3-24, 1983.

BOENTE, A.N.P.; MORÉ, J.D. Um modelo *Fuzzy* para avaliação da satisfação dos gerentes de projetos de produtos de software de uma fundação pública estadual. In: ENCONTRO NACIONAL DE ENGENHARIA DE PRODUÇÃO, 29., Salvador, 2009.

CORDEIRO, A.G.; MOLL, R.N. Pesquisa de satisfação de usuários de software de gestão hospitalar utilizando os critérios da ISO 9126. In: X CONGRESSO BRASILEIRO DE INFORMÁTICA EM SAÚDE, 10., *Anais*... Florianópolis, 2006.

CORDEIRO, A.G.; FREITAS, A.L.P. O cenário atual da qualidade de software. In: SIMPÓSIO DE ENGENHARIA DE PRODUÇÃO, 15., *Anais*... Bauru, 2008.

COUGHLAN, J.; MACREDIE, R.D. Effective communication in requirements elicitation: a comparison of methodologies. *Requirements Engineering,* v. 7, p. 47–60, 2002.

DANTAS, C.R.; MURTA, L.G.P.; WERNER, C.M.L. Consistent evolution of UML models by automatic detection of change traces. In: EIGHTH INTERNATIONAL WORKSHOP ON PRINCIPLES OF SOFTWARE EVOLUTION (IWPSE'05), 2005. Proceedings..., 2005, p. 14-147.

DENNING, P.J. What is software quality. *Communications of ACM,* v. 35, n. 1, 1992.

DUKE, C.R.; MOUNT, A.S. Rediscovering performance-importance analysis of products. *Journal of Product & Brand Management,* v. 5, n. 2, p. 43-54, 1996.

DUKE, C.R. *Learning outcomes: comparing student perceptions of skill level and importance*. *Journal of Marketing Education*, v. 24, p. 203-217, 2002.

FREITAS, A.L.P.; MANHÃES, N. R. C.; COZENDEY, M. I. Emprego do SERVQUAL na avaliação da qualidade de serviços de tecnologia da informação: uma análise experimental. In: ENEGEP, 26., 2006. *Anais*... 2006. p. 1-8.

FREITAS, A.L.P.; RODRIGUES, S.G.; COSTA, H.G.. Emprego de uma abordagem multicritério para classificação do desempenho de instituições de ensino superior. *Ensaio: aval.pol.públ.Educ.* v.17, n. 65, p. 655-674, 2009.

FREITAS, A.L.P.; BOLSANELLO, F.M.C.; VIANA, N.R.N.G. Avaliação da qualidade de serviços de uma biblioteca universitária: um estudo de caso utilizando o modelo Servqual. *Ciência da Informação*, v. 37, n. 3, p. 88-102, 2008.

GAROFALAKIS, J.; STEFANI A.; STEFANIS, V. A framework for the quality evaluation of B2C M-Commerce services. *International Journal of Handheld Computing Research.,* v. 2, n. 3, p. 73-91, 2011.

GOUSIOS, G.; KARAKOIDAS, V.; STROGGYLOS, K.; LOURIDAS, P.; VLACHOS, V.; SPINELLIS, D. Software quality assessment of open source software. In: PANHELLENIC CONFERENCE ON INFORMATICS, PCI 2007, 11th., Athens, 2007. p. 303–315

HUDSON, S.; HUDSON, P.; MILLER, G.A. The measurement of service quality in the tour operating sector: a methodological comparison. *Journal of Travel Research*, v. 42, p. 305-312, 2004.

JANZEN, D.S.; SAIEDIAN, H. Does test-driven development really improve software design quality? *IEEE Software*, v. 25, n 2, p. 77-84, 2008.

JUNG, H. Validating the external quality subcharacteristics of software products according to ISO/IEC 9126. *Computer Standards & Interfaces*, v. 29, p. 653-661, 2007.

KARLSSON, J.; RYAN, K. Supporting the selection of Software Requirements. In: INTERNATIONAL WORKSHOP ON SOFTWARE SPECIFICATION AND DESIGN (IWSSD '96), *8th. Proceedings* … 1996, p. 146-149.

KARLSSON, J.; WOHLIN, C.; REGNELL, B. An evaluation of methods for prioritizing software requirements. *Information and Software Technology*. v.39, p. 939-947, 1998.

KOSCIANSKI, A.; SOARES, M.S. *Qualidade de software: aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software*. 2. ed., São Paulo: Novatec, 2007. 395 p.

LARMAN, C. *Utilizando UML e padrões: uma introdução à análise e ao projeto orientado a objetos e ao Processo Unificado.* 2. ed., Porto Alegre: Bookman, 2004. 607p.

LARSEN, T.J. A multilevel explanation of end-user computing satisfaction with an enterprise resource planning system within an international manufacturing organization. *Computers in Industry*, v. 60, n. 9, p. 657-668, 2009.

LEEWORTHY, V.R.; WILEY, P.C. *Importance and satisfaction ratings by recreating visitors to the Florida Keys/ Key West*. The University of Georgia, 1996. 27p.

LEFFINGWELL, D.; WIDRIG, D. *Managing Software Requirements*. Addison Wesley, 1999. 528 p.

MAGAL, S.R.; LEVENBURG, N.M. Using importance-performance analysis to evaluate e-business strategies among small businesses. In: HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES, 38th, 2005. *Proceedings*…

MAGALHÃES, A.L.C. A Garantia da qualidade e o SQA: sujeito que ajuda e sujeito que atrapalha. *ProQualiti – Qualidade na produção de software*, v. 2, n.2, p. 9-14, 2006.

MAIDEN, N.A.M.; RUGG, G. ACRE: selecting methods for requirements acquisition. *Software Engineering Journal*, v. 11, n. 3, p. 183-192, 1996.

MALHOTRA, N. *Pesquisa de Marketing: uma orientação aplicada*. 4. ed., Porto Alegre: Bookman, 2006. 720p.

MARTILLA, J.A., JAMES, J.C. Importance-performance analysis. *Journal of Marketing*, n.9, p.41-77, 1977.

PRESSMAN, R.S. *Engenharia de Software*. 5. ed., Rio de Janeiro: McGraw-Hill, 2002. 843p.

PUNTER, T.; KUSTERS, R.; TRIENEKENS, J.; BEMELMANS, T.; BROMBACHER, A. The w-process for software product evaluation: a method for goal-oriented implementation of the ISO 14598 standard. *Software Quality Journal*, v. 12, p. 137–158, 2004.

ROBERTSON, S.; ROBERTSON, J. *Mastering the requirements process*. 2. ed., Addison Wesley Professional. 592 p. 2006.

ROSQVIST, T.; KOSKELA, M.; HARJU, H. Software quality evaluation based on expert judgement. *Software Quality Journal*. v.11, n. 39–55, 2003.

SADRAEI, E.; AURUM, A.; BEYDOUN, G.; PAECH, B. A field study of the requirements engineering practice in Australian software industry. *Requirements Engineering*. v. 12, p.145-162, 2007.

SAINI, R.; DUBEY, S. K.; RANA, A. Analytical study of maintainability models for quality Evaluation. *Indian Journal of Computer Science and Engineering*, v. 2, n. 3, p. 449-454, 2011.

SKOK, W.; KOPHAMEL, A.; RICHARDSON, I. Diagnosing information systems success: importance–performance maps in the health club industry. *Information & Management,* v. 38, p. 409-419, 2001.

SRIVASTAVA, P. R.; KUMAR. S; SINGH, A.P.; RAGHURAMA, G. Software testing effort: an assessment through fuzzy criteria approach. *Journal of Uncertain Systems*. v.5, n. 3, p. 183-201, 2011.

YOUNG, R.R. *The requirements engineering handbook*. Boston: Artech House, 2004. 251 p.

APÊNDICE 1

Questionário de priorização de requisitos

UENF Mestrado PPGEP-UENF
Universidade Estadual do Norte Fluminense Darcy Ribeiro

FUNDAÇÃO Dr. João Barcellos Martins — HOSPITAL FERREIRA MACHADO

Esta pesquisa tem como objetivo a priorização dos requisitos relativos ao sistema que está sendo desenvolvido para os setores de Departamento Pessoal, Recursos Humanos e Seguran ça e Medicina do trabalho. Atrav és dessa pesquisa pretendemos desenvolver o software para atender da melhor forma possível às necessidades dos setores envolvidos. **Por isso, esteja atento às questões para definir suas respostas**

Função: _____ Tempo de Experiência no setor: _____

1) Na tabela a seguir observe os requisitos, características que o sistema deve possuir. Marque na tabela a importância relativa a cada requisito.

	REQUISITOS	Grau de Importância				
		Nada Importante	Pouco Importante	Neutro	Importante	Muito Importante
1	**Controle de Freqüência dos Funcionários**					
1.1	Registro de freqüência	1	2	3	4	5
1.2	Registro de informações de escalas e horários de trabalho	1	2	3	4	5
1.3	Emissão de relatório de freqüência	1	2	3	4	5
2	**Manutenção dos dados pessoais**					
2.1	Controle das informações pessoais dos funcionários	1	2	3	4	5
2.2	Registro de deficiências apresentadas pelos funcionários	1	2	3	4	5
3	**Manutenção dos dados funcionais**					
3.1	Emissão de crachás	1	2	3	4	5
3.2	Manutenção de informações relativas à convocação	1	2	3	4	5
3.3	Controle de Alocação de funcionários em unidade e setores	1	2	3	4	5
4	**Manutenção dos Afastamentos**					
4.1	Controle de Férias	1	2	3	4	5
4.2	Controle de Afastamentos como Licenças Médicas e INSS	1	2	3	4	5
5	**Movimentação de Documentos**					
5.1	Entrada e saída de carteiras de trabalho e previdência social	1	2	3	4	5
5.2	Solicitação e entrega de declarações	1	2	3	4	5

2) Considere que você tem 100 pontos, distribua-os pelos requisitos apresentados de acordo com a necessidade referente a cada requisito. A soma dos pontos distribuídos deve totalizar 100 pontos.

	REQUISITOS	PONTOS
1	**Controle de Freqüência dos Funcionários**	
1.1	Registro de freqüência	
1.2	Registro de informações de escalas e horários de trabalho	
1.3	Emissão de relatório de freqüência	
2	**Manutenção dos dados pessoais**	
2.1	Controle das informações pessoais dos funcionários	
2.2	Registro de deficiências apresentadas pelos funcionários	
3	**Manutenção dos dados funcionais**	
3.1	Emissão de crachás	
3.2	Manutenção de informações relativas à convocação	
3.3	Controle de Alocação de funcionários em unidade e setores	
4	**Manutenção dos Afastamentos**	
4.1	Controle de Férias	
4.2	Controle de Afastamentos como Licenças Médicas e INSS	
5	**Movimentação de Documentos**	
5.1	Entrada e saída de carteiras de trabalho e previdência social	
5.2	Solicitação e entrega de declarações	
	TOTAL:	100

3) Informe outros requisitos que você julgue relevantes e que não foram encontrados nas listas acima:

_____

_____

APÊNDICE 2

Questionário de avaliação de desempenho com base nos requisitos

UENF Mestrado PPGEP-UENF
Universidade Estadual do Norte Fluminense Darcy Ribeiro

Fundação Dr. João Barcellos Martins — Ferreira Machado

Esta pesquisa tem como objetivo a avaliação do sistema que está sendo desenvolvido para os setores de Departamento Pessoal, Recursos Humanos e Segurança e Medicina do trabalho. Através dessa pesquisa pretendemos verificar os requisitos do software que precisam ser alterados para atender da melhor forma possível às necessidades dos setores envolvidos. **Por isso, esteja atento às questões para definir suas respostas**.

Função: _____ Tempo de Experiência no setor: _____

1) Na tabela a seguir observe os requisitos, características que foram consideradas necessárias no sistema. Marque na tabela o grau de desempenho relativo a cada requisito.

		Grau de Desempenho				
	REQUISITOS	Muito Ruim	Ruim	Neutro	Bom	Muito Bom
1	**Controle de Freqüência dos Funcionários**					
1.1	Registro de freqüência	1	2	3	4	5
1.2	Registro de informações de escalas e horários de trabalho	1	2	3	4	5
1.3	Emissão de relatório de freqüência	1	2	3	4	5
2	**Manutenção dos dados pessoais**					
2.1	Controle das informações pessoais dos funcionários	1	2	3	4	5
2.2	Registro de deficiências apresentadas pelos funcionários	1	2	3	4	5
3	**Manutenção dos dados funcionais**					
3.1	Emissão de crachás	1	2	3	4	5
3.2	Manutenção de informações relativas à convocação	1	2	3	4	5
3.3	Controle de Alocação de funcionários em unidade e setores	1	2	3	4	5
4	**Manutenção dos Afastamentos**					
4.1	Controle de Férias	1	2	3	4	5
4.2	Controle de Afastamentos como Licenças Médicas e INSS	1	2	3	4	5

2) Forneça um pontuação de 0 a 100 de acordo com o funcionamento de cada requisito do sistema. Os requisitos que você considera que estão funcionando melhor devem receber uma pontuação maior.

EXEMPLO: ***Requisito A***	Muito Ruim ├----┼----┼----┼----┼-X--┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100
Registro de Freqüência	Muito Ruim ├----┼----┼----┼----┼----┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100
Registro de informações de escalas e horários de trabalho	Muito Ruim ├----┼----┼----┼----┼----┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100
Emissão de relatório de freqüência	Muito Ruim ├----┼----┼----┼----┼----┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100
Controle das informações pessoais dos funcionários	Muito Ruim ├----┼----┼----┼----┼----┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100
Registro de deficiências apresentadas pelos funcionários	Muito Ruim ├----┼----┼----┼----┼----┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100
Emissão de crachás	Muito Ruim ├----┼----┼----┼----┼----┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100
Manutenção de informações relativas à convocação	Muito Ruim ├----┼----┼----┼----┼----┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100
Controle de Alocação de funcionários em unidade e setores	Muito Ruim ├----┼----┼----┼----┼----┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100
Controle de Férias	Muito Ruim ├----┼----┼----┼----┼----┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100
Controle de Afastamentos como Licenças Médicas e INSS	Muito Ruim ├----┼----┼----┼----┼----┼----┼----┼----┼----┼----┼----┤ Muito Bom 0 10 20 30 40 50 60 70 80 90 100

3) Comentários e observações (*use o verso da folha, se for preciso*):

_____

_____