

Requisitos Não Funcionais no Desenvolvimento de Software Educacional: Um Estudo Utilizando Grounded Theory

Alanna C. C. Monteiro
Universidade Estadual da Paraíba
Brasil
alannacoelho@hotmail.com

Vivianne de Queiroz Leal
Ministério Público da Paraíba
Brasil
vivianne@mp.pb.gov.br

Reinaldo C. de M. Gomes
Universidade Federal de Campina
Grande
Brasil
reinaldo@dsc.ufcg.edu.br

Luciana de Q. L. Gomes
Universidade Estadual da Paraíba
Brasil
luciana@cct.uepb.edu.br

ABSTRACT

Educational practices has been suffering significant changes in terms of resources employed to conduct the classes. In order to make possible the modernization of classes taught is essential that the Software Engineering, particularly the requirements engineering, possesses techniques to subsidize the construction process of educational software. This research presents an application of Grounded Theory methodology in discovering requirements more appropriate to the development of educational software, from the perspective of a degree in computer science. As a result, we obtained 12 non-functional requirements related to the development of educational software.

RESUMO

As práticas educativas vem sofrendo mudanças significativas no tocante aos recursos utilizados para a realização das aulas. No sentido de viabilizar a modernização das aulas ministradas é indispensável que a Engenharia de Software, particularmente a engenharia de requisitos, disponha de técnicas para subsidiar o processo de construção dos softwares educacionais. Este estudo traz, portanto, uma aplicação da metodologia *Grounded Theory* na descoberta de requisitos mais apropriados ao desenvolvimento de softwares educacionais, na perspectiva do licenciado em informática. Como resultado, foram obtidos 12 requisitos não funcionais relacionados ao desenvolvimento de softwares educacionais.

Categories and Subject Descriptors

D.2 [Software Engineering]: Requirements/Specifications.

General Terms

Management, Documentation, Human Factors.

Keywords

Requisitos não funcionais, Grounded theory, Desenvolvimento de software educacional, Engenharia de Requisitos.

1. INTRODUÇÃO

A evolução e difusão da tecnologia vem despertando a necessidade de utilização de softwares voltados para a educação, tanto em meio acadêmico ao disseminar os conteúdos ministrados conferindo aos mesmos um maior alcance, como em meio corporativo enquanto instrumento facilitador em treinamentos e capacitações.

Desta feita, analisar, comparar e apresentar adequadamente os requisitos de um sistema torna-se de extrema importância, em particular no desenvolvimento de softwares educativos que possuem caráter peculiar diante da diversidade dos profissionais envolvidos. É imprescindível, portanto, a utilização de um processo bem definido e fundamentado nos princípios da engenharia de requisitos, a qual é definida por Sommerville [9] como o processo de descobrir, analisar, documentar e verificar os serviços fornecidos pelo sistema e suas restrições operacionais.

Os desenvolvedores de software têm procurado a melhor forma de levantar os requisitos de um sistema e, dessa forma, diversas técnicas e métodos foram criados para garantir que um sistema atenda às necessidades e expectativas dos clientes [1], [2], [3]. Contudo, uma questão interessante, que vale a pena ser ressaltada, envolvendo o desenvolvimento de aplicações educativas está na especificidade dos requisitos não funcionais de usuários, que, muitas vezes, parecem pouco integralizados no desenvolvimento [4].

Este estudo pretende apresentar a engenharia de requisitos de software no cenário de desenvolvimento de softwares educacionais. Para tanto, busca-se na teoria fundamentada ou *Grounded Theory* [10] um método para extrair resultados mais significativos da perspectiva do cliente ou usuário do software, com relação aos requisitos necessários a softwares educacionais.

Na próxima seção, são apresentadas as atividades da Engenharia de Requisitos, com a descrição das etapas do processo e das técnicas de levantamento e análise de requisitos. A seção seguinte trata do método de pesquisa *Grounded Theory* e como este método é aplicado neste estudo. Seguem-se a esta os resultados da análise diante do método escolhido e as considerações finais, bem como os possíveis trabalhos a serem desenvolvidos.

2. A ENGENHARIA DE REQUISITOS E O SOFTWARE EDUCACIONAL

A engenharia de requisitos é um processo que envolve todas as atividades exigidas para criar e manter o documento de requisitos de sistema [9]. Assim sendo, um dos grandes desafios para o engenheiro de software é justamente compreender os requisitos do problema, uma vez que nem sempre os clientes/usuários tem o domínio do que aqueles representam. É importante detectar previamente qual será o impacto do produto de software sobre o negócio e como os usuários finais interagirão com o sistema. Nesta linha, Batista [1] destaca que a definição dos requisitos não

é um processo matemático e que há fatores organizacionais, técnicos e sociais envolvidos.

Há uma tendência comum entre os diversos estudiosos da área em classificar em etapas a atividade de requisitos, dentro do processo de desenvolvimento como um todo [6]. Apesar de cada um deles incluir uma característica pessoal em sua classificação, esta atividade envolve basicamente as etapas de concepção e levantamento, seguindo-se com a elaboração de artefatos para negociação com o cliente bem como para dirimir conflitos que porventura ocorram, a especificação ou formalização em representação matemática ou não, mas que forneça aos projetistas subsídios para um bom projeto de sistema. Por fim, a validação quanto aos atributos de qualidade da completeza e consistência, além da gestão de necessidades e mudanças.

O processo de concepção e levantamento inicia-se com a identificação dos interessados e reconhecimento dos seus pontos de vista com a finalidade de confrontar as inconsistências e apontar áreas em comum. No que se refere ao projeto de um software educacional, a falta de clareza ou equívoco nesta fase pode ser demasiadamente danoso ao ciclo de desenvolvimento em sua totalidade, principalmente quando o público-alvo contempla pessoas que usarão a ferramenta para adquirir conhecimentos basilares para a sua formação. Fracassos podem ser evitados incluindo uma equipe multidisciplinar, de pedagogos, professores, dentre outros profissionais, e os próprios alunos, neste momento. A presença dos principais interessados instiga também a atividade de comunicação e possibilita algumas personalizações.

Neste sentido, Mandel [8] aponta o processo de concepção como um dos motivos da dificuldade em produzir um software educacional de qualidade, dadas as diferenças significativas entre as representações que os envolvidos tem acerca dos processos de ensino e aprendizagem.

Na análise, o escopo inicialmente estabelecido é refinado e aperfeiçoado em detalhes. Tal análise proporciona a construção de representações formais da informação, o que fomenta a elaboração de um projeto procedimental, de arquitetura e de dados.

Vale a pena ressaltar que documentar o que é analisado e desenvolvido em cada fase é fundamental para que problemas ou mudanças futuras sejam mais facilmente administrados, facilitando, portanto, o trabalho dos desenvolvedores.

A documentação colabora também para o gerenciamento dos requisitos, ou seja, para a compreensão e controle das mudanças. O planejamento desta fase deverá se iniciar ao mesmo tempo em que o levantamento inicial, para que uma regra ou restrição uma vez alterada não implique em modificações bruscas em relação aos demais que com aquele se relaciona. Gerenciar também tem o fim de identificar se os requisitos elencados foram cumpridos.

3. GROUNDED THEORY

Grounded Theory ou teoria fundamentada é uma metodologia qualitativa cujo modelo de investigação objetiva criar uma relação entre a teoria e a realidade estudada, com especial destaque ao papel ativo do investigador [10]. Proposta por Glaser e Strauss, se fundamenta no método da descoberta através das condições contextuais em que os fenômenos ocorrem. As teorias devem ser delineadas com base nos dados que lhes dão origem, numa abordagem interativa de coleta e análise dos dados, organizados em uma sequência que tende para uma maior complexidade e integração. Trata-se, portanto, de um processo indutivo do conhecimento.

O processo se inicia com a definição do problema de investigação. Nesta etapa, é importante que o pesquisador tome nota de todas as questões, definindo os limites do fenômeno. A ideia central é que se elaborem questões abertas que permitam uma análise flexível das respostas, e de igual forma, um estudo em profundidade comprometido com os limites do problema. No entanto, as próprias questões são mutáveis e passíveis de análise, e não se esgotam no início da investigação. Neste sentido se apresenta o princípio fundamental da análise no âmbito da *Grounded Theory*: o método da comparação constante. Através deste, se desenrola um movimento contínuo, que transita entre a análise do investigador e um retorno constante aos dados, até que o processo fique saturado.

Com o desenvolvimento do método, outras questões vão sendo formuladas e validadas, evoluindo de questões abertas para questões progressivamente mais focadas e voltadas para o cenário do estudo.

A construção da amostra se dá no decorrer da própria análise, onde questões e ideias vão se completando e tomando feição representativa das nuances do fenômeno, sendo direcionada aos dados [5]. Assim sendo, é coerente ir analisando os dados coletados, através de entrevistas, por exemplo, à medida que são realizadas, até que as categorias se estabilizem e que os novos eventos em nada agreguem ao estudo.

Em termos de codificação, é possível identificar três tipos de abordagens: aberta, coaxial e seletiva. A codificação aberta é o processo de decompor os dados em unidades de análise, examiná-los, compará-los, conceituá-los e categorizá-los. Isso ajuda a centrar a atenção no fenômeno e abre caminho para a construção do conhecimento indutivo.

Inicialmente se decompõe os dados em algo que represente o fenômeno – uma observação, uma pergunta, um acontecimento – e o questiona, nomeando e conceituando as respostas que vão surgindo. Em seguida, os conceitos construídos devem ser agrupados em categorias, de acordo com as relações de similaridade entre eles. Logo, a codificação aberta consiste na identificação de categorias, bem como na definição flexível de propriedades e dimensões.

A codificação axial é um procedimento posterior à codificação aberta, através do qual os dados já conceituados são recategorizados, indo além das propriedades e dimensões uma vez definidos. Nesse caso, elege-se uma categoria como central e com as demais se estabelece relações de subordinação, sejam elas causais, contextuais ou intervenientes.

Já a codificação seletiva é o processo de escolher a categoria central, em torno da qual as outras serão integradas. Mesmo não diferindo muito da codificação axial, é com essa abordagem que os dados são refinados e representados em alto nível de abstração, sendo nesta fase que o investigador constrói a narrativa descritiva do fenômeno em estudo, integrando, por fim, as categorias em forma de teoria.

4. APLICAÇÃO DO MÉTODO

O levantamento dos dados foi realizado através de entrevista guiada por roteiro, aplicada a vinte alunos do curso de Licenciatura em Informática. O corpo docente do referido curso é, em sua maioria, composto por Bacharéis em Ciência da Computação, com pós-graduação (mestrado e/ou doutorado) na área de Engenharia de Software.

O estudo realizado teve como objetivo levantar, através da ótica dos alunos, quais são os requisitos não funcionais mais importantes para o desenvolvimento de softwares educacionais. Para este fim, os respondentes foram questionados com relação a sua própria experiência de uso de softwares educacionais.

A coleta dos dados foi realizada de forma aberta e anônima. Cada um dos alunos respondeu as seguintes questões: Você já usou um jogo, ou algum outro software, para auxiliar o aprendizado de alguma disciplina? O que é mais importante para você quando vai utilizar um software? Que elementos são essenciais para que um software seja utilizado por você? O que falta aos softwares educacionais para que sejam melhor compreendidos? Como você imaginaria um software para a disciplina que você tem dificuldade? Qual a sua opinião sobre o uso de um software com muitos componentes de interface?

Para utilizar o método *Grounded Theory* é necessário examinar cada entrevista, mais de uma vez e por mais de um pesquisador, de maneira independente. Esta atividade é realizada para evitar vies e identificar unidades de análise para a realização da codificação aberta que procura estabelecer categorias conceituais. A Tabela 1 apresenta as categorias obtidas (requisitos não funcionais) na codificação aberta e exemplos de unidades de análise (trechos de respostas dos alunos) utilizadas para a obtenção de cada categoria.

Tabela 1. Requisitos não funcionais obtidos a partir da codificação aberta

Id	Requisito Não Funcional	Unidade de Análise
1	Facilidade de uso	“E se eu sei que o software é complicado de utilizar, difícil de usar, geralmente eu procuro o que é mais simples pra poder aprender e depois passar para um mais avançado.”
2	Confiabilidade	“Se eu sei que eu tenho que software que tem problema, que vai bugar no meio da apresentação do que esteja usando com ele, eu já não vou utilizar.”
3	Adequação de conteúdo	“Para que ele (o software educacional) seja melhor compreendido você tem que pensar no contexto que você vai utilizar, você tem que pensar no seu público. O contexto envolve público, estrutura que você tem e o seu próprio conhecimento daquele software.”
4	Interatividade	“Quando tem muito, muito, muito (componentes de interface) é ruim. Tudo em excesso é veneno. Quando tem recurso gráfico, principalmente interatividade. Acho que a interface tem que criar interatividade, boa interatividade, fácil interatividade.”
5	Clareza de conteúdo	“Acho que tem que ter linguagem clara e objetiva, que ajude o usuário a entender o software, se o conteúdo está relacionado com o assunto que você está buscando aprender.”
6	Aprendizagem	“E com o uso do software eu pude compreender melhor o conteúdo e

		esclarecer as dúvidas que ainda tinha. E, além disso, ainda tive maior facilidade para memorizar os assuntos, pois as animações e a interface do software contribuíram para um melhor entendimento.”
7	Encadeamento de ideias	“Além disso, a sequência do conteúdo é fundamental importância porque para que o entendimento não seja quebrado.”
8	Relação entre a teoria e a prática	“O software que utilizei permitiu que conhecesse mais a fundo o conteúdo estudado, porque ele conciliava a realidade e a teoria e isso permitiu fixar melhor o conteúdo.”
9	Usabilidade	“A parte da interface gráfica, a parte que os alunos veem mesmo, facilita. Para eles olharem e ver como faz, como manusear.”
10	Documentação	“Tipo assim, o que eu vejo dificuldades nas pessoas de entender um software no caso, tem muita gente que faz ele e joga pra galera aprender sozinho. Eu não acho isso certo. Eu prefiro assim, colocar um tutorial, uma instrução para a galera poder usar com mais facilidade, é melhor.”
11	Presença de elementos gráficos	“Não só eu, mas muitos alunos aprendem mais associando imagens com textos, ou com vídeos.”
12	Atratividade	“Quanto mais cores, mais desenhos, mais prende a atenção do aluno.”

Ao aplicar *Grounded Theory*, o processo investigativo de construção de uma teoria é realizado quando o conhecimento é agregado a uma teoria por meio da análise e construção das relações entre categorias e subcategorias. Essas categorias e subcategorias devem ser revisitadas e testadas novamente nos dados [10]. A Tabela 2 apresenta o resultado da relação delineada entre os requisitos não funcionais obtidos.

Tabela 2. Grupos de requisitos obtidos através da codificação axial

Id	Grupo de Requisitos	Requisitos Não Funcionais
1	Requisitos genéricos	Facilidade de uso, Confiabilidade, Usabilidade, Documentação
2	Requisitos educacionais	Adequação de conteúdo, clareza de conteúdo, Aprendizagem, Encadeamento de ideias, Relação entre a teoria e a prática
3	Requisitos lúdicos	Interatividade, Presença de elementos gráficos, Atratividade

Os requisitos apresentados na Tabela 2 foram agrupados de acordo com suas características centrais em:

- **Requisitos genéricos:** requisitos não funcionais usuais, necessários a qualquer tipo de software.

- **Requisitos educacionais:** requisitos específicos, relacionados com o aprendizado do aluno e assimilação de conteúdos.
- **Requisitos lúdicos:** requisitos relacionados com a liberdade e espontaneidade de ação.

Assim, ao realizar a codificação seletiva para este estudo, os grupos de requisitos obtidos (Requisitos genéricos, Requisitos educacionais e Requisitos lúdicos) podem ser interpretados como requisitos fundamentais ao desenvolvimento de softwares educacionais, sob a ótica do cliente ou usuário. Cada um dos grupos de requisitos não funcionais, e os próprios requisitos não funcionais levantados, representam deficiências a serem sanadas no desenvolvimento de softwares educacionais, relacionadas à qualidade do produto.

Considerando isto e dado que o método não foi executado em sua integralidade, uma conclusão parcial sobre o estudo realizado até então poderia apontar como central a categoria: **Restrições para o desenvolvimento de softwares educacionais relacionadas ao uso do produto**. Esta conclusão, ainda que parcial, é levantada, pois ao desenvolver um software, o que se busca com requisitos não funcionais é o entendimento das questões relacionadas ao uso da aplicação, às tecnologias empregadas no seu desenvolvimento, à infraestrutura de suporte e à qualidade, características estas que delimitam restrições para a elaboração do produto de software.

5. CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Este estudo apresentou, através do uso de *Grounded Theory* e sob a perspectiva de alunos do curso de Licenciatura em Informática, requisitos não funcionais relacionados ao desenvolvimento de softwares educacionais.

Observando a complexidade exigida em um software educacional, recomenda-se a formação de uma equipe multidisciplinar para a construção do mesmo, pois neste, a fase de levantamento de requisitos se mostra ainda mais complexa quando comparada aos softwares convencionais, por envolver *stakeholders* de diferentes áreas de conhecimento [7]. Isto também se aplica ao levantamento de requisitos não funcionais, e este trabalho apresentou 12 deles como contribuição para esta atividade.

Grounded Theory estabelece a necessidade de coletas e análises intercaladas. Contudo, o presente estudo realizou apenas uma coleta de dados, o que significa que a saturação teórica ainda não foi alcançada. Considerando este fato, ainda não foi possível validar as categorias obtidas, tampouco propor uma teoria, mas os resultados alcançados servem de insumo para as próximas execuções de coleta e análise intercaladas. Além disso, os requisitos não funcionais obtidos podem ser utilizados para identificar outros requisitos não funcionais adequados ao desenvolvimento de softwares educacionais.

Assim, enumeram-se como trabalhos futuros a finalização do estudo utilizando *Grounded Theory*, o que culminará em uma teoria, e a avaliação dos requisitos não funcionais obtidos como resultados da pesquisa.

6. REFERENCIAS

- [1] BATISTA, Edinelson Aparecido. (2003) Uma Taxonomia Facetada para Técnicas de Elicitação de Requisitos. Dissertação de Mestrado. São Paulo: UNICAMP.
- [2] BORTOLI, L. A. de & PRICE, A. M. de A. (2000) O Uso de Workflow para Apoiar a Elicitação de Requisitos. WER 2000, disponível em < http://wer.inf.puc-rio.br/WERpapers/artigos/artigos_WER00/bortoli.pdf> Acesso em 05/03/2010.
- [3] CRUZ NETO, Genésio Gomes da & GOMES, Alex Sandro & TEDESCO, Patricia. (2003) Elicitação de Requisitos de Sistemas Colaborativos de Aprendizagem Centrada na Atividade de Grupo. In: SBIE 2003. Novembro, Rio de Janeiro.
- [4] GOMES, Alex Sandro & TEDESCO, Patricia Azevedo & CASTRO FILHO, José Aires de. (2003) Ambientes de Aprendizagem em Matemática e Ciências. In: Edla Maria Faust Ramos; Marta Costa Rosatelli; Raul Sydney Wazlawick. (Org.). Informática na Escola: um Olhar Multidisciplinar. 01 ed. Fortaleza: Editora Universidade Federal do Ceará, v. 1, p. 108-135.
- [5] FERNANDES, Eugénia M., MAIA & Ângela. (2001) Grounded theory. In FERNANDES, Eugénia M.; ALMEIDA Leandro S., ed. lit. Métodos e Técnicas de Avaliação: Contributos para a Prática e Investigação Psicológicas. Braga: Universidade do Minho. Centro de Estudos em Educação e Psicologia.
- [6] KOTONYA, Gerald & SOMMERVILLE, Ian.: Requirements Engineering: Processes and Techniques (1998) John Wiley & Sons, Ltd.
- [7] LACERDA, Rafael de Alencar (2007) Proposta de um modelo para análise de requisitos de software educativo. Dissertação de mestrado. Brasília: UnB, 2007.
- [8] MANDEL, Theo. (1997) The Elements of User Interface. New York: John Wiley and Sons.
- [9] SOMMERVILLE, Ian (2007) Engenharia de software. 8ª edição. São Paulo: Pearson Alison Wesley.
- [10] STRAUSS, Anselm & CORBIN, Juliet (2008) Pesquisa Qualitativa: Técnicas e procedimentos para o desenvolvimento de teoria fundamentada. 2ª edição. Artmed.