Avaliação da Qualidade dos Aspectos em Documentos de Requisitos

Ricardo Argenton Ramos, Jaelson F. B. Castro, Alexandre M. L. Vasconcelos
Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Av. Prof. Luiz Freire S/N – Recife – PE – Brasil 50732-970, +55 81 3271 8430
e-mail: [rar2, jbc, amlv]@cin.ufpe.br

Resumo

A área de desenvolvimento orientado a aspectos esta cada vez mais ampla, abrangendo todas as fases no processo de desenvolvimento de software. Pesquisas promissoras estão sendo feitas na fase de requisitos cujo o propósito é separar no documento de requisitos os aspectos (requisitos não funcionais) que estão espalhados e entrelaçados aos requisitos funcionais. Contudo, não se tem conhecimento de proposta para avaliar a qualidade da definição dos aspectos que são elicitados na fase de requisitos. Portanto, esse artigo propõe uma metodologia e um conjunto de métricas para a avaliação da qualidade de aspectos em documentos de requisitos orientado a aspectos. Esta metodologia foi elaborada a partir de trabalhos prévios de medição da qualidade em documentos de requisitos e da definição de métricas para avaliar o código fonte orientado a aspectos.

Abstract

The aspect oriented development area is widening its scope, addressing all phases of the software development process. Indeed, recent proposals have looked at the requirements stage, trying to separate at the requirements documents the aspects (non functional requirements) that are spread and tangled with functional requirements. However, we are not aware of any proposals do evaluate the quality of these aspects discover during the requirements elicitation. This paper proposes a methodology and a set of metrics to assist evaluation of the quality of aspect oriented requirements documents. The proposal is based on previous experience on requirements document evaluation as well as the definition of aspect oriented source code metrics.

1. Introdução

O uso de processos de desenvolvimento de software de modo inadequado gera grandes custos nas fases de manutenção deste software. De acordo com Pressman [20], o custo da correção de erros na fase de projeto é cerca de três a seis vezes mais alto do que na fase de definição de requisitos. Este custo aumenta ainda mais quando a correção de erros é realizada em fases mais avançadas no processo de desenvolvimento.

No sentido de melhorar o entendimento e conseqüentemente a qualidade do documento de requisitos, desta maneira antecipando a correção de possíveis erros, alguns trabalhos propõem a utilização de técnicas de separação de interesses (do inglês *concerns*) utilizando conceitos do Desenvolvimento Orientado a Aspectos (DOA) na fase de requisitos [22], cuja a idéia principal é que documentos de requisitos são melhores entendidos quando os seus requisitos funcionais e não funcionais estiverem especificados separadamente. Além dessa separação de interesses as pesquisas nessa área fornecem algumas técnicas para modelagem e ainda como e onde posteriormente combinar o que foi separado [22, 27].

Apesar das prováveis facilidades que o DOA sugere para a fase de requisitos, não se encontra na literatura trabalhos que avaliem qualitativamente um documento de requisitos orientado a aspectos. Santa´Ana e outros [25, 26] avaliam qualitativamente e quantitativamente os benefícios de implementações de padrões de projeto em sistemas

orientados a aspectos e orientados a objetos, porém em seus trabalhos os documentos de requisitos não são avaliados.

Na área da avaliação de qualidade de documentos de requisitos, Reis [24] propõe uma metodologia para medir a qualidade desses documentos no domínio de sistemas Web, baseando-se em árvores que descrevem as características de um determinado requisito não funcional. Ainda nesse contexto, Durán [8] elabora métricas que verificam a presença de erros em casos de uso. Ambos trabalhos foram utilizados como base para a elaboração das métricas descritas nesse artigo.

O objetivo desse artigo é fornecer métricas que ajude a garantir que os aspectos modelados em um documento de requisitos direcionado a casos de uso e orientado a aspectos estejam completos (segundo os aspectos avaliados) e que a complexidade desse documento em relação ao tamanho, acoplamento e a separação de interesses seja medida.

Na Seção 2 estão sendo relatados alguns trabalhos na área da programação orientada a aspectos na fase de requisitos, na Seção 3 são apresentadas metodologias de medição de qualidade, na Seção 4 é descrito como foi realizado o estudo de caso. A proposta de uma metodologia e métricas para avaliar a qualidade dos aspectos em um documento de requisitos direcionado a casos de uso é relatada na Seção 5, na Seção 6 são comentadas algumas restrições no estudo de caso e os comentários finais e trabalhos futuros são apresentados na Seção 7.

2. Orientação a Aspectos na Fase de Requisitos

Produzir software com qualidade e num menor tempo possível é um dos principais objetivos da Engenharia de Software [20]. Para alcançar tal objetivo, essa área de pesquisa estabeleceu alguns princípios que devem ser aplicados ao longo do processo de desenvolvimento e nos artefatos de software: separação de Interesses, modularidade, rigor e formalidade, incrementalidade, abstração, generalidade e antecipação de mudanças [11].

O principal propósito do princípio da separação de Interesses é reduzir o raciocínio para uma quantidade factível [6]. Esse princípio estabelece que um problema deve ser decomposto em unidades menores, claramente separadas, de maneira que cada uma delas represente um único interesse [1]. A idéia básica é manipular com um interesse do sistema de cada vez. Como conseqüência, a aplicação desse princípio possibilita [11]:

- diminuir a complexidade do desenvolvimento de software, concentrando-se em diferentes interesses separadamente;
- raciocinar sobre diferentes interesses de maneira relativamente independente;
- dividir esforços e separar responsabilidades entre membros da equipe de desenvolvimento; e
- melhorar a modularidade de artefatos de software.

O desenvolvimento orientado a aspectos surgiu como uma das áreas de pesquisas promissoras dentro da separação de interesses, onde a idéia principal é que sistemas são melhores entendidos se os aspectos (requisitos não funcionais) ficarem separados da parte funcional do software [9, 10, 15, 16].

Elrad e outros [9] afirmam que o desenvolvimento orientado a aspectos consiste na separação dos interesses de um sistema em unidades modulares e posterior composição (*weaving*) desses módulos em um sistema completo. Os interesses podem variar de noções de alto nível, como segurança e qualidade de serviço [22, 23], a noções de baixo nível, como sincronização, manipulação de *buffers* de memória, tratamento de exceções, persistência entre outros [5, 21] .

A Programação Orientada a Aspectos (POA) trata os interesses que entrecortam as classes de modo análogo ao que a programação orientada a objetos faz para o

encapsulamento e a herança. Ou seja, provê mecanismos que explicitamente capturam a estrutura de entrecorte, alcançando assim, os benefícios de melhor modularidade, tais como: código mais simples, mais fácil de manter e alterar e, conseqüentemente, aumento da reusabilidade [15, 5, 21].

Grande esforços dos pesquisadores tem sido em direção da implementação de sistemas orientados a aspectos [5, 9, 10, 21], porém recentemente algumas pesquisas surgem utilizando o desenvolvimento orientado a aspectos na fase de requisitos [4, 18, 22, 23], onde tem-se o propósito de separar no documento de requisitos os aspectos (requisitos não funcionais) que estão espalhados e entrelaçados aos requisitos funcionais.

Rashid e outros [23] propuseram um modelo de processo genérico, denominado AORE (*Aspect-Oriented Requirements Engineering*), para separar interesses transversais no nível de requisitos. No modelo AORE, o termo "interesse" é utilizado num sentido mais restrito, correspondendo a um requisito não-funcional de alto nível de abstração (ex: segurança, compatibilidade).

Utilizando como base o processo de desenvolvimento unificado [13], Sousa [28] realiza uma adaptação desse processo para o desenvolvimento orientado a aspectos, onde o principal foco desta pesquisa é o desenvolvimento guiado por casos de uso [14], possibilitando a separação de interesses transversais nas fases de requisitos, análise e projeto utilizando fundamentos da programação orientada a aspectos.

Segundo a proposta de Sousa [28], as unidades bases (que podem ser afetadas por aspectos) correspondem às unidades de decomposição comumente utilizadas em cada um dos fluxo de atividades do processo unificado; casos de uso no fluxo de requisitos; classes de análise no fluxo de análise; e classes de projeto no fluxo de projeto. Sendo que os possíveis pontos de junção referem-se a elementos nessas unidades que representam pontos de execução: no fluxo de requisitos pode-se especificá-los em função dos eventos (ou passos) nos cenários de um caso de uso; no de análise, em termos de mensagens trocadas entre objetos de análise; e no de projeto, em termos de operações de classes.

Partindo do princípio de que aspectos são encapsulados nas mesmas unidades de decomposição existentes em cada um dos fluxos considerados, Sousa [28] cria um estereotipo especial para cada fase, como por exemplo, caso de uso aspectual, classe de análise aspectual, para indicar que o comportamento do aspecto deverá ser inserido em alguma parte das unidades que ele afeta. O termo *crosscuts* refere-se ao relacionamento entre uma unidade aspectual e uma unidade afetada por ela. Uma tabela denominada de tabela de composição armazena para cada aspecto identificado e em cada fluxo, informações relativas à composição entre um aspecto e a(s) unidade(s) que ele afeta.

Para validar seu trabalho, Sousa [28] desenvolve com sua abordagem um sistema de Internet Banking Orientado a Aspectos. Seguindo a abordagem de Sousa o diagrama de casos de uso aspectual, mostrado na Figura 1, apresenta os casos de uso (requisitos funcionais), como proposto pela UML [13], e os casos de uso aspectuais, que neste caso modularizam o requisito não funcional de segurança. Esses casos de uso aspectuais estão modelados no formato de losangos e nos relacionamentos que indica que o caso de uso aspectual entrecorta um (ou mais) caso de uso, existe o estereotipo *crosscuts*.

Para cada caso de uso aspectual existe uma tabela que o descreve e existe ainda uma segunda tabela que define onde e quando o caso de uso aspectual entrecorta o caso de uso base.

Para o contexto desse trabalho o termo interesse refere-se a um requisito não funcional, sendo que esse pode ser implementado em um ou mais aspectos. Portanto, no diagrama de casos de uso aspectual um interesse pode ser modelado em um ou mais casos de uso aspectuais.

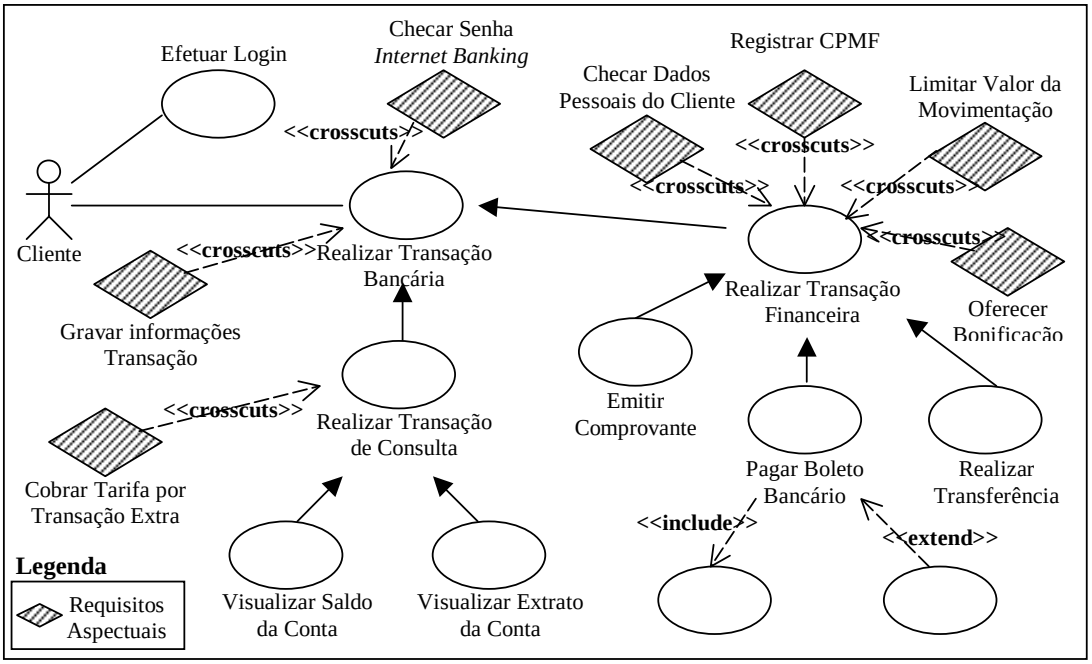

Figura 1 - Diagrama aspectual de casos de uso do sistema Internet Banking.

A pesar das vantagens propostas pelo desenvolvimento orientado a aspectos na fase de requisitos, como um documento de requisitos melhor organizado e de mais fácil manutenção, não existe uma metodologia que avalie a qualidade desses documentos. Na Seção seguinte são apresentados alguns trabalhos relacionados na área de verificação de qualidade, dois relacionados com a qualidade na fase de requisitos [8, 24] e um na área da programação orientada a aspectos na fase de código [26].

3. Metodologias para Verificação da Qualidade

A comunidade de engenharia de software vem adaptando padrões existentes como a ISO 9126 [12] para que desta maneira se permita fornecer um suporte para que haja conformidade e assim possibilitando elaborar métricas para medir a qualidade de um produto de software [12]. A medição de um software é importante para garantir a qualidade do produto, avaliar a produtividade das pessoas que o produzem e formar uma linha base para estimativas, portanto são muitas as vantagens de se poder medir a qualidade e corrigir seus erros, inconsistências e saber sua completitude, porém quanto mais cedo nas fases de desenvolvimento do software for descoberto os erros, menos custoso é o seu processo de correção [20]. Por esse motivo a medição da qualidade de um produto de software na fase de requisitos tem sido uma área bastante promissora.

As atividades que asseguram a qualidade em documentos de requisitos são divididas em três, como sumarizada na Tabela 1 [8].

Tabela 1 - Atividades que asseguram a qualidade em documentos de requisitos.

Atividade	Objetivo Principal
Análise	Identifica conflitos em requisitos.
Verificação	Detecta defeitos em requisitos.
Validação	Certifica que os requisitos são consistentes com as intenções dos clientes e usuários.

Nesse artigo são focados 2 trabalhos que fazem a verificação da qualidade, um que verifica erros em casos de uso [8] e segundo que verificara a completitude de requisitos não funcionais [24]. Um terceiro trabalho relatado nesse artigo realiza a verificação de

erros no código fonte de sistemas orientados a aspectos [26]. Esses trabalhos são apresentados nas subseções seguintes.

3.1. Verificação da Qualidade na Fase de Requisitos

No contexto de verificação de qualidade na fase de requisitos, Durán [8] propõe uma ferramenta denominada REM (*REquirements Manager* - Gerenciador de Requisitos), que dá apoio a verificação de requisitos que estão armazenados em XML [30], em seu trabalho são elaboradas heurísticas que tratam de alguns problemas recorrentes em documentos de requisitos, tais como: ambigüidade, completitude, rastreabilidade. A ferramenta REM também dá apoio a verificação de casos de uso, desta maneira foram elaboradas algumas métricas para verificar a qualidade em casos de uso.

As métricas elaboradas no trabalho de Durán [citar] medem fatores de complexidade em caso de uso como o número de passos de um caso de uso, passos condicionais, exceções e o número de vezes que um caso de uso é incluído ou estendido em outros casos de uso. Essas métricas foram utilizadas como base na elaboração das métricas desse artigo.

Algumas métricas como a relatada anteriormente apenas fornecem mecanismos para se medir o tamanho, ou a complexidade, não fornecendo mecanismos para indicar se esse ou aquele valor obtido esta bom ou ruim. Porém existem métricas que auxiliam o engenheiro de software a fazer uma avaliação dando um escore final para um determinado fator que foi medido, como é o caso da metodologia apresentada a seguir.

Reis [24] propõe uma metodologia para verificação da qualidade de aplicações Web na fase de requisitos denominada de REQE (*Requirements Engineering Quality Evaluation*). Esta metodologia se propõe a verificar a qualidade de aplicações Web na fase inicial do processo de desenvolvimento, tendo assim como benefício à possibilidade de descobrir erros de uma forma antecipada.

A aplicação da metodologia REQE consiste em cinco fases:

1. Representação das características, subcaracterísticas e atributos de qualidades;

Nesta primeira fase serão elicitados e especificados os requisitos de qualidade, culminando com uma árvore que, hierarquicamente, lista todas as características, subcaracterísticas e atributos de qualidade, capazes de modelar a qualidade segundo as necessidades do usuário. Cada atributo de qualidade deve ser medido por meio de um escore de conformidade. Este escore verifica se o produto atenderá ou não (com base no documento de requisitos) àquele atributo de qualidade.

2. Especificação descritiva da árvore de característica, subcaracterísticas e atributos de qualidade;

Nesta fase, todos os nós da Árvore de Características, Subcaracterísticas e Atributos de Qualidade (ACSAQ) devem ser especificados de forma descritiva. É preciso descrever cada nó detalhadamente, a fim de dirimir eventuais dúvidas a cerca de seus significados. A título de documentação, Reis [24] propõe que para cada característica, subcaracterística e atributo de qualidade, deve ser preenchida uma planilha de informação, baseada na planilha em [19].

3. Associação de pesos aos nós;

Nesta fase da metodologia REQE deve-se determinar o grau de importância dos nós no contexto do domínio da aplicação, por meio da associação de peso a cada um dos nós. Primeiramente atribui-se pesos às características, em seguida as subcaracterísticas e ao atributos de qualidade.

Reis decidiu adotar o valor 10 (dez) para ser distribuído entre os filhos de cada nó da ACSAQ. Primeiramente deve-se distribuir 10 (dez) pontos entre as características presentes na árvore. São permitidos apenas números inteiros positivos. Essa pontuação vai

representar o grau de importância daquela característica, na qualidade global da aplicação Web.

4. Associação de escores aos atributos de qualidade;

Ao iniciar esta etapa já se obteve a árvore de características, subcaracterísticas e atributos de qualidade estruturada, juntamente com as planilhas de informações preenchidas e os pesos associados a todos os nós das árvore (que identificam a importância relativa de cada item). Para a atribuição de valores, Reis [24] propõe a seguinte escala de conformidade, baseado em [1] e [3]:

Tabela 2 - Tabela de conformidade x escore.

Conformidade	Escore
Não atende o atributo de qualidade	0
Atende com sérias restrições	3
Atende parcialmente	6,5
Atende	10

Para cada atributo de qualidade da ACSQA deve ser associado um escore, seguindo a escala da Tabela 2. Caso o atributo não seja previsto no documento de requisitos, deve receber a conformidade "não atende" (escore 0), pois como o documento não prevê, é considerado que não atende o atributo de qualidade, conseqüentemente, o produto final dificilmente atenderá. Nesta situação, o atributo ausente deverá ser anotado na seção de recomendações do documento de requisitos, para que seja analisado posteriormente a razão da sua omissão. Deste modo a metodologia proposta por Reis[24] esta contribuindo para a melhoria da qualidade da aplicação no momento em que identificam prováveis falhas.

5. Cálculo geral.

O objetivo desta fase é efetuar o calculo que consolida todos os escores de conformidade associados, levando em consideração os pesos dos nós. Ao final desta fase a avaliação da qualidade é concluída e a possibilidade de corrigir os erros identificados (atributos de qualidade mal avaliados, isto é mal especificados). Neste estágio do processo, todos os atributos estão com seus escores associados. A nota de cada nó-pai é calculada através dos escores e dos pesos dos nós filhos.

Seguindo a fórmula para o cálculo da nota de nó pai, Figura 2, o peso de cada atributo filho é multiplicado pelo seu escore, todos esses valores são adicionados, e a soma deve ser dividida por 10 (dez), resultando na nota do pai. Esta divisão por 10 (dez) é uma espécie de normalização, uma vez que cada nó teve o valor 10 (dez) dividido entre os pesos de seus nós filhos.

$$\textit{Nota-Nó-Pai} = \left(\sum_{i=1}^{n} (\textit{peso} * \textit{escore}) \right) / 10$$

Onde i varia do primeiro ao último filho.

Figura 2 - Fórmula para cálculo da nota de nó pai.

Na Figura 3, a subcaracterística SubcaracterísticaA1 foi calculada com base nos pesos e escores de conformidade de seus filhos AtributoA11 eAtributoA12. A nota da subcaracterística SubcaracteríticaB1 também foi calculada através dos pesos e escores de conformidade de seus filhos. Já a nota da característica CaracterísticaB será calculada por meio dos pesos e das notas das subcaracterísticas SubcaracterísticaB1 e SubcaracterísticaB2. A nota da característica CaracterísticaA será calculada através dos pesos e das notas das subcaracterísticas SubcaracterísticaA1, SubcaracterísticaA2 e do atributo de qualidade AtributoA3.

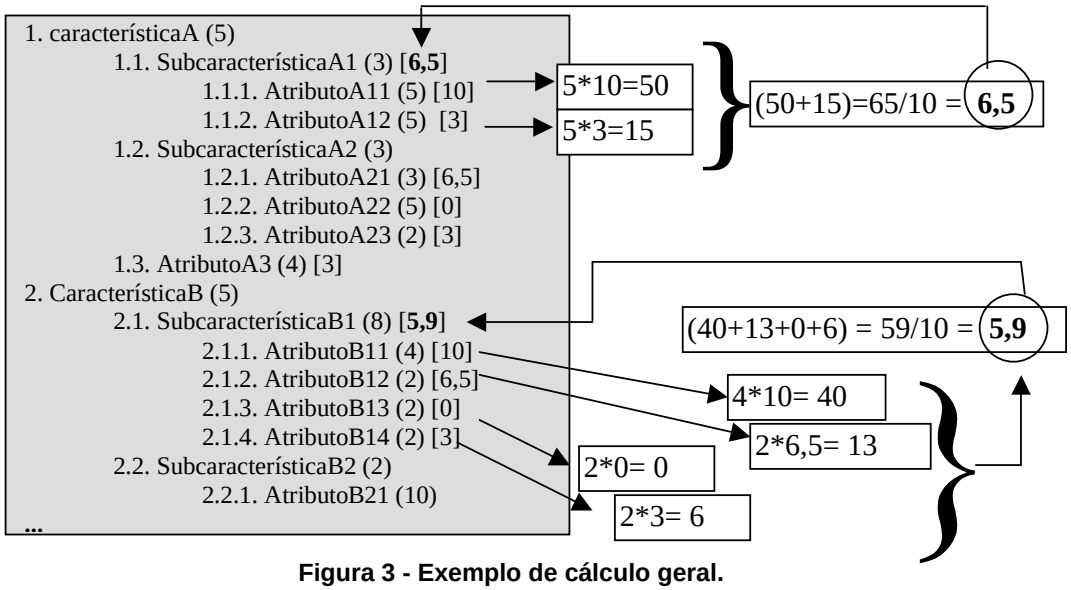

Figura 3 - Exemplo de cálculo geral.

Essas métricas serviram como base para a elaboração das métricas propostas nesse artigo, pois como o estudo de caso realizado foi em um documento de requisitos orientado a aspectos e direcionado a casos de uso, foi necessário adaptar métricas utilizadas em casos de uso e em orientação a aspectos como as que serão apresentadas na Seção seguinte.

3.2. Métricas para Avaliar Qualitativamente e Quantitativamente o Código Fonte de Sistemas Orientados a Aspectos

O tamanho do código fonte indica a quantidade de esforço necessária para entender os componentes do sistema. A separação de interesses propicia o aumento do entendimento do sistema, pois o código fica bem organizado e possibilitando a identificação de cada um dos seus componentes (aspectos e classes) [9, 10, 15, 16, 21]. Sant´ana [26] elabora um conjunto de métricas que ao serem aplicadas no código fonte de um sistema orientado a aspectos, fornece informações relacionadas a separação de interesses, acoplamento e tamanho. Um conjunto dessas métricas é apresentado na Tabela 3.

Tabela 3 - Métricas para avaliar a qualidade do código fonte orientado a aspectos.

	Métricas	Definição
Separação de Interesses	Difusão de um Interesse em Componentes (CDC)	Conta o número de classes e aspectos cuja a finalidade principal é contribuir para a implementação de um interesse e o número de outras classes e aspectos que os acessa.
	Difusão de um Interesse em Operações (CDO)	Conta o número de métodos e “advices” cuja a finalidade principal é contribuir para a implementação de um interesse e o número de outros métodos e “advices” que os acessa.
	Difusão de um Interesse em LOC (CDLOC)	Conta o número de pontos de transição para cada interesse através das linhas de código. Pontos de transição são pontos no código onde existe “Mudança de Interesse” (do inglês *concern switch*).
Acoplamento	Acoplamento entre Componentes (DIT)	Conta o número de outras classes e aspectos que são acoplados a uma classe ou um aspecto.
	Profundidade da Árvore de Herança	Conta o quão profundo é a declaração de uma hierarquia de herança de uma classe ou de um aspecto.
Tamanho	Linha de Código (LOC)	Conta as linhas de código
	Número de Atributos (NOA)	Conta o número de atributos de cada classe ou aspecto
	Largura de Operações por Componentes (WOC)	Conta o número de métodos e “advices” de cada classe ou aspectos e o número de parâmetros de cada um.

Em seu trabalho Sant´ana [26] realiza um estudo comparativo entre implementações de padrões de projeto com o paradigma orientado a objetos e com o orientado a aspectos. Os resultados da aplicação das métricas mostraram que em alguns casos a separação de interesses melhorou a qualidade da implementação. No entanto, alguns padrões ficaram mais acoplados, mais complexos e com mais número de LOC.

4. Estudo de Caso

Existem duas técnicas básicas para avaliar uma abordagem: experimentos e estudos de caso. A realização de experimentos provê uma maneira de avaliação baseada em comparação direta, permitindo a pesquisadores investigar qual o impacto da tecnologia no processo de maneira detalhada [28]. Já a realização de estudos de caso está mais preocupada em avaliar os benefícios de uma abordagem de forma a verificar se as mudanças no processo proporcionam o resultado desejado [17]. Nesse artigo a técnica de avaliação escolhida foi o estudo de caso, pois se esta interessado em avaliar o benefício da aplicação de uma metodologia de medição de qualidade em um outro contexto, diferente para o qual a metodologia foi elaborada.

O sistema escolhido para o estudo de caso foi um sistema bancário disponível na Internet, também denominado de sistema de Internet Banking. O principal objetivo desse tipo de sistema é permitir que clientes realizem transações bancárias através da Internet. Os serviços cobertos pelo sistema Internet Banking são: visualizar saldo, visualizar extrato de conta, fazer transferência e realizar pagamento de boleto bancário. O requisito não funcional que será o foco da aplicação da metodologia REQE [24] é o de segurança. Sendo que esse foi o interesse modularizado em aspectos pela abordagem de Sousa [28], Figura 1.

O estudo de caso foi elaborado com o foco de se avaliar a aplicabilidade da abordagem REQE [24] em um sistema orientado a aspectos, onde apenas um engenheiro de software foi envolvido na medição da qualidade. A seguir o estudo de caso é detalhado seguindo as fases da metodologia REQE sem adaptações na metodologia.

4.1. Fase 1 – Representação das Características, Subcaracterísticas e Atributos de Qualidade

Esta fase é dividida em dois passos, sendo que o no primeiro é feita a escolha do domínio da aplicação e são traçados os possíveis perfis dos usuários interessados. No segundo passo é especificada a Árvore de Características, Subcaracterísticas e Atributos de Qualidade (ACSAQ).

Passo 1 – Escolha do Domínio da Aplicação e do Perfil do Interessado;

Domínio da aplicação: Site de um Sistema Bancário

Perfil do Interessado: Cliente do Banco, visitante intencional.

Na metodologia de Reis [24] esse Passo é realizado com auxílio da engenharia de domínio, porém nesse estudo de caso o domínio da aplicação e o perfil do interessado foram obtidos com base nas descrições do trabalho de Sousa [28].

Passo 2 – Especificação da árvore de Características, Subcaracterísticas e Atributos de Qualidade.

Seguindo a metodologia de Reis [24], para cada domínio de aplicação e perfil do interessado deve-se selecionar os principais requisitos não funcionais. Quando se tratar de uma medição real isso deve ser feito por meio de questionários orientados aos clientes do banco e discussões entre as partes interessadas. Nesse estudo de caso foi feita uma adaptação do requisito de segurança a partir de catálogos oferecidos pelo *framework* NFR [6]. A árvore de Características, Subcaracterísticas e Atributos de Qualidade pode ser vista na Figura 5.

4.2. Fase 2 - Especificação Descritiva da Árvore de Características, Subcaracterísticas e Atributos de Qualidade

Nesse artigo não serão apresentadas todas as planilhas de subcaracterísticas e atributos de qualidade, devido ao volume dessa documentação. A Figura 4. mostra a planilha da característica de Segurança. O engenheiro de software deve estar ciente de que é preciso descrever cada nó detalhadamente, a fim de dirimir eventuais dúvidas a cerca de seus significados [24].

PLANILHA DE CARACTERÍSTICA

Título: Segurança
Código: 1
Quantidade de Filhos: 4
Subcaracterísticas: (1.1.) Integridade, (1.2.) Disponibilidade, (1.3.) Confidencialidade, (1.4.) Confiabilidade
Atributos: não há.
Definição: Proteger a Informação de diversas ameaças para garantir a continuidade dos negócios, a integridade e a disponibilidade da mesma.
Peso: 10 (este peso é por ser somente essa característica que esta sendo medida)
Escore: a ser calculado posteriormente

Figura 4 - Planilha preenchida da característica Segurança.

4.3. Fase 3 – Associação de Pesos aos Nós

Nesta fase deve-se atribuir pesos aos nós, sendo que quanto mais importante for a característica, a subcaracterística ou atributo de qualidade, maior deve ser o seu peso em relação aos nós do seu nível. Esses pesos devem ser associados pelos avaliadores. A fim de comparação de aplicações Web, aplicações do mesmo domínio devem ser submetidas à mesma distribuição de pesos. Nesse estudo de caso os pesos foram atribuídos seguindo a ordem de importância dada no *framework* NFR proposto por Chung e outros [6]. A Figura 5 mostra (entre parênteses) como ficou distribuído os pesos de cada nó na ACSAQ.

4.4. Fase 4 – Associação de Escores aos Atributos

Nessa fase os avaliadores devem procurar pelos atributos descritos na árvore no documento de requisitos, sendo que para cada atributo da árvore deve ser atribuído um escore seguindo a tabela de conformidade mostrada na Tabela 2.

Para buscar no documento de requisitos do sistema orientado a aspecto o atributo 1.1.2.2. Limitar Valor de Movimentação, somente foi necessário procurar entres os casos de uso aspectuais, Figura 1, e para saber quais são as descrições deste caso de uso foi necessário olhar em sua especificação, Tabela 4. Desta maneira o engenheiro de requisitos pode saber que este atributo não esta espalhado por outras partes do documento de requisitos, porém este atributo pode afetar outros casos de uso, isso pode ser conhecido quando for observada a Tabela de Composição, que mostra os casos de uso afetados por um determinado caso de uso aspectual, Tabela 5.

Tabela 4 - Especificação do caso de uso aspectual Limitar Valor de Movimentação.

OPERACIONALIZACAO #03: Limitar valor de movimentação	
INFORMAÇÕES CARACTERÍSTICAS	
Objetivo	Verificar limite do valor de transações financeiras para diminuir riscos de fraudes.
Pré-condições	Valor da transação é dado como entrada.
Ator Primário	Cliente do banco, titular da conta.
CENÁRIO PRINCIPAL DE SUCESSO	
Passo	Ação
1	O sistema verifica se o valor da transação supera limite.
2	O sistema retorna o resultado da verificação

Tabela 5- Tabela de composição Limitar Valor de Movimentação.

REQUISITO ASPECTUAL: AR #07 LIMITAR VALOR DE MOVIMENTAÇÃO				
Caso de uso Afetado	Condição (Opcional)	Regra de Composição	Ponto de Junção	Informações Adicionais
UC #04 Realizar transferência	-	Wrap	3	Se limiteOk então execute(ponto junção) Senão exibe(msgErro)
UC #05 Pagar boleto bancário	-	Wrap	4	Se limiteOk então execute(ponto junção) Senão exibe(msgErro)

Como o caso de uso aspectual atende totalmente ao atributo de qualidade, para esse é dado o escore 10. Pois foi observado que o aspecto contém descrito o início (dado de entrada), meio (ação de processamento) e fim (resultado esperado). Para os outros atributos o processo de pontuação foi semelhante, sempre seguindo a conformidade mostrada na Tabela 2, e buscando os atributos de segurança somente nos aspectos. A Figura 5 mostra como ficou a ACSAQ no final das atividades da fase 4.

1. Segurança (10)
- 1.1. Integridade (3) **[]**
  - 1.1.1Completude (4) **[]**
    - 1.1.1.1. Protocolo de correção de erros (10) **[0]**
  - 1.1.2. Precisão (6) **[]**
    - 1.2.1. Instalar um *FireWall* (6) **[0]**
    - 1.2.2. Limitar valor de movimentação (4) **[10]**
- 1.2. Disponibilidade (2) **[]**
  - 1.2.1. Espelhar o banco de dados (6) **[3]**
  - 1.2.2. Duplicar o Servidor (4) **[3]**
- 1.3. Confidencialidade (3) **[]**
  - 1.3.1. Criptografar dados (4) **[6,5]**
  - 1.3.2. Autorizar acesso (6) **[]**
    - 1.3.2.1. Identificar usuário (5)**[]**
      - 1.3.2.1.1. Número da conta (5) **[10]**
      - 1.3.2.1.2. Número da agência (5) **[10]**
    - 1.3.2.2. Autenticar usuário (5) **[]**
      - 1.3.2.2.1. Checar senha (5) **[10]**
      - 1.3.2.2.2. Checar dados pessoais do cliente (4) **[10]**
      - 1.3.2.2.3. Checar senha adicional (1) **[0]**
- 1.4. Confiabilidade (2) **[]**
  - 1.4.1. Arquivo de Log (10) **[6,5]**

Figura 5 - Árvore para a característica de Segurança em um sistema de banco pela Internet com pesos e escores.

Os nós que não são atributos estão com os colchetes vazios, pois terão suas notas calculadas na próxima fase, seguindo a metodologia.

4.5. Fase 5 – Cálculo Geral

Nessa última fase todos os outros nós (subcaracterísticas e características) terão uma nota associada. Seguindo a formula para o calculo do nó pai, mostrada anteriormente na Figura 2, serão calculadas e atribuídas as notas como mostrado na Figura 6.

A nota final do requisito não funcional de segurança para este sistema Internet Banking, é de **4,645**, que foi calculada com a divisão por 10 da soma dos resultados da multiplicação dos pesos e escores de cada subcaracterística desse aspecto de segurança. Essa medição serviu para identificar uma série de falhas (na verdade requisitos que não foram levados em consideração no documento de requisitos ou mal especificados), que podem ser corrigidas no início do processo de desenvolvimento, acarretando a construção de um produto de melhor qualidade.

1. Segurança (10) [4,645] (3 * 2,4 + 2 * 3 + 3 * 6,75 +2 * 6,5)= 46,45/10 = **4,645**
1.1. Integridade (3) **[2,4]** (4 * 0 + 6*4)= 24/10 = **2,4**
1.1.1Completude (4) **[0]** (10 * 0)= 0/10 = **0**
1.1.1.1. Protocolo de correção de erros (10) **[0]**
1.1.2. Precisão (6) **[4]** (6 * 0 + 4*10)= 40/10 = **4**
1.1.2.1. Instalar um *FireWall* (6) **[0]**
1.1.2.2. Limitar valor de movimentação (4) **[10]**
1.2. Disponibilidade (2) **[3]** (6 * 3 + 4*3)= 30/10 = **3**
1.2.1. Espelhar o banco de dados (6) **[3]**
1.2.2. Duplicar o Servidor (4) **[3]**
1.3. Confidencialidade (3) **[6,75]** (4 * 6,5 + 6*9,5)= 67,5/10 = **6,75**
1.3.1. Criptografar dados (4) **[6,5]**
1.3.2. Autorizar acesso (6) **[9,5]** (5 * 10 + 5*9)= 95/10 = **9,5**
1.3.2.1. Identificar usuário (5)**[10]** (5 * 10 + 5*10)= 100/10 = **10**
1.3.2.1.1. Número da conta (5) **[10]**
1.3.2.1.2. Número da agência (5) **[10]**
1.3.2.2. Autenticar usuário (5) **[9]** (5 * 10 + 4*10 + 1 * 0)= 90/10 = **9**
1.3.2.2.1. Checar senha (5) **[10]**
1.3.2.2.2. Checar dados pessoais do cliente (4) **[10]**
1.3.2.2.3. Checar senha adicional (1) **[0]**
1.4. Confiabilidade (2) **[6,5]** (10 * 6,5)= 65/10 = **6,5**
1.4.1. Arquivo de Log (10) **[6,5]**

Figura 6 - ACSAQ do sistema Internet Banking com pesos e escores.

5. Uma Proposta de Uma Metodologia e Métricas para a Avaliação da Qualidade de Aspectos em Documentos de Requisitos

As métricas apresentadas nessa Seção tem como objetivo fornecer uma base para que um engenheiro de software consiga medir a qualidade de um interesses que foi modelado em casos de uso aspectuais em um documento de requisitos direcionado a casos de uso e orientado a aspectos. As cinco fases da metodologia REQE [24] foram adaptadas para apoiar a medição dos aspectos incluídos no diagrama de casos de uso. Além das fases que foram adaptadas, uma nova fase é criada para realizar medidas relacionadas a separação de interesses e ao tamanho e acoplamento dos aspectos. A adaptação da metodologia REQE consiste em seis fases, que são descritas a seguir:

5.1. Instanciação das Árvores de Interesse, Sub-Interesses e Atributos de Qualidades

Na metodologia REQE [24] em seu formato original nesta primeira fase serão elicitados e especificados os requisitos de qualidade, culminando com uma árvore que, hierarquicamente, lista todas as características, subcaracterísticas e atributos de qualidade, capazes de modelar a qualidade segundo as necessidades do usuário. Com a realização do estudo de caso notou-se que cada característica representa um interesse e conseqüentemente cada subcaracterística um sub-interesse. Decidiu-se então elaborar um catalogo de árvores de Interesses e Sub-Interesses contendo alguns modelos (*templates*) dos principais interesses conhecidos no nível de requisitos. Portanto, quando o engenheiro de software desejar utilizar alguma das árvores de interesses e sub-interesses do catálogo, ele apenas instancia essa árvore para o domínio de sua aplicação definindo os atributos característicos do domínio, utilizando para isso uma engenharia de domínio, como proposto pela metodologia REQE.

A primeira versão do Catálogo de Interesses e Sub-Interesses contém 5 interesses: Segurança, Usabilidade, Desempenho, Custos e Confiabilidade. Essas árvores foram elaboradas com base nos trabalhos de Olsina [19] (Usabilidade e Confiabilidade) e no *framework* RNF [6] (Segurança, Desempenho e Custos). A Figura 7 mostra como é a árvore de Interesses e Sub-Interesses de Desempenho.

1. Desempenho
    1.1. Espaço
    1.2. Tempo
        1.2.1. Vazão
        1.2.2. Tempo de Resposta
            1.2.2.1. Tempo de Acesso
            1.2.2.2. Tempo de Resposta

Figura 7 - Árvore de interesse e sub-interesses de desempenho.

O engenheiro de software deverá aumentar ou diminuir a quantidade de características e subcaracterísticas das árvores dependendo do domínio de sua aplicação. Desta maneira também deverão ser incluídos os atributos de qualidade que são específicos para cada domínio de aplicação.

5.2. Especificação Descritiva das Árvores de Interesse, Sub-Interesses e Atributos de Qualidades

Seguindo a metodologia REQE, na segunda fase é necessário descrever cada nó das árvores detalhadamente , a fim de dirimir eventuais dúvidas a cerca de seus significados. A única mudança proposta para essa fase é que ao invés se preencher uma planilha para cada característica e subcaracterítica agora será preenchida uma para cada interesse e sub-interesse.

5.3. Associação de Pesos aos Nós

Nesta fase da metodologia REQE deve-se determinar o grau de importância dos nós no contexto do domínio da aplicação, por meio da associação de peso a cada um dos nós. Primeiramente atribui-se pesos às características, em seguida as subcaracterísticas e ao atributos de qualidade. Assim como a fase anterior, as únicas mudanças realizadas é a alteração de características e subcaracterísticas para interesses e sub-interesses.

5.4. Associação de Escores aos Atributos de Qualidade

Nesta etapa é necessário que se procure por cada atributo de qualidade nos casos de uso aspectuais. Se encontrado o atributo em um caso de uso, deve-se observar sua descrição afim de saber seu grau de completitude. Uma descrição de caso de uso geralmente é organizada com início (dado de entrada), meio (ação de processamento) e fim (resultado esperado) [13, 14], portanto infere-se que essa organização facilite a obtenção do grau de completitude de um atributo de qualidade.

Assim como na metodologia REQE os escores para cada atributo de qualidade deve ser associado um escore, seguindo a escala da Tabela 2. Caso o atributo não seja previsto no documento de requisitos, deve receber a conformidade “não atende” (escore 0), pois como o documento não prevê, é considerado que não atende o atributo de qualidade, conseqüentemente, o produto final dificilmente atenderá.

5.5. Medição da Separação de Interesses, Tamanho e Acoplamento dos Casos de Uso Aspectuais

A metodologia REQE, verifica a qualidade de um documento de requisitos em relação a sua completitude. Porém com a utilização do Desenvolvimento Orientada a Aspectos para a organização de documentos de requisitos, surgem outros fatores que a metodologia não previa, como: o quão separado esta um determinado interesse ou então, qual o grau de

acoplamento de um aspecto. Tendo essas motivações decidiu-se elaborar algumas métricas baseadas em trabalhos tanto de verificação de qualidade em casos de uso [8] quanto de medição da qualidade do código fonte orientado a aspectos [25, 26].

Esta fase também poderá ser utilizada separadamente da metodologia, quando a intenção for somente medir fatores relacionados com a separação de interesses. Porém essa nova fase foi adicionada na metodologia para que se possa elaborar uma base de conhecimento e saber qual foi o impacto da utilização de aspectos na modelagem de um determinado interesse. Essa base de conhecimento servirá para realizar comparações com outros documentos de requisitos e assim poder inferir qual desses esta com um determinado interesse melhor modularizado.

As métricas são divididas em três tipos:

1 - Separação de interesses:

- Número de casos de uso aspectual que modelam um determinado interesse;
- Número de passos nas descrições dos casos de uso aspectual que descrevem um determinado interesse.

2 - Tamanho:

- Número de passos nas descrições de cada caso de uso aspectual;
- Número de passos condicionais de cada caso de uso aspectual;
- Número de exceções de cada caso de uso aspectual.

3 - Acoplamento:

- Número de casos de uso (aspectual ou não) que são incluídos ou estendidos em um determinado caso de uso aspectual;
- Número de casos de uso (aspectual ou não) que são entrecortados por um determinado caso de uso aspectual.

O primeiro tipo de métrica verifica o quão separado esta um determinado interesse, ou seja, em quantos casos de uso aspectuais estão modelados um determinado interesse e quantos passos foram necessários para descrever o interesse. Os dados referentes a quantidade de passos podem ser obtidos nas tabelas criadas para descrever cada aspecto, como a mostrada na Seção 4, Tabela 4. Esse primeiro tipo de métrica terá um valor para cada interesse, por exemplo: “Utilizando as métricas de separação de interesses foi possível verificar que o interesse de Segurança foi modularizado por 7 (sete) casos de uso aspectuais e descrito em 27 (vinte e sete) passos”.

O segundo conjunto de métricas são relacionadas ao tamanho de cada caso de uso aspectual, indicando fatores como o número de passos, passos condicionais e exceções. Esses números podem ser obtidos nas tabelas que descrevem os casos de uso aspectuais. Essas métricas terão um valor único para cada caso de uso aspectual, por exemplo: “Utilizando as métricas de tamanho foi possível verificar que o caso de uso aspectual Limitar Valor da Movimentação contém 2 (dois) passos e 0 (zero) exceções e passos condicionais”.

As métricas relacionadas com o acoplamento verifica a quantidade de relacionamentos tanto de entrecorte quanto de extensão e inclusão existentes em cada caso de uso aspectual. Os números de inclusão e extensão podem ser obtidos na tabela que descreve o caso de uso aspectual. Porém o número de unidades que são entrecortadas por um caso de uso aspectual devem ser obtidos na tabela de composição, como a mostrada na Seção 4, Tabela 5.

5.6. Calculo Geral

Essa última fase não foi alterada, permanecendo da mesma maneira como proposta pela metodologia REQE [24]. Porém o engenheiro de software deve estar ciente que a

qualidade agora é medida por interesse que estão modularizados em casos de uso aspectuais, podendo assim optar por medir a qualidade de somente um interesse ou de quantos mais desejar.

6. Restrições do Estudo de Caso

O foco do estudo de caso realizado é a avaliação da aplicabilidade de uma metodologia de medição da qualidade de documento de requisitos utilizada em sistemas orientado a objetos para medir a qualidade de um documento de requisitos orientado a aspectos. Sendo assim alguns métodos sugeridos pela metodologia não foram seguidos, podendo assim ter o resultado final desta medição afetado, porém não afetando a avaliação de sua aplicabilidade.

A principal restrição a ser comentada é que a medição foi feita por somente um avaliador, sendo que o ideal seria o apoio desde engenheiros de domínio até as opiniões de clientes. Por esse motivo, algumas adaptações foram realizadas, como para gerar a árvore de características, subcaracterísticas e atributos de qualidade, utilizou-se como base o framework NFR proposto por Chung e outros [6]. Utilizando como base ainda o mesmo framework NFR foram atribuídas as pontuações quanto a importância de cada subcaracterística.

Para o estudo de caso foi escolhida a metodologia proposta por Reis [24], pois esta apóia a medição da qualidade de um interesse (requisito não funcional) na fase de requisitos. Porém outras metodologias semelhantes, conhecidas por medir a qualidade de requisitos não funcionais, não foram incluídas nesse artigo. Sendo assim um trabalho futuro a ser realizado é a comparação de outras metodologias para a mesma finalidade.

7. Considerações Finais e Trabalhos Futuros

Este artigo apresentou um estudo de caso aplicando uma metodologia proposta para medir a qualidade de documento de requisitos de aplicações Web em um documento de requisitos direcionado a casos de uso e orientado a aspectos com a intenção de observar quais são os benefícios dessa combinação e o que poderia ser proposto para melhorar essa avaliação.

A proposta da adaptação da metodologia REQE [24] para avaliar a qualidade de documentos de requisitos direcionados a casos de uso e orientados a aspectos apresentada nesse artigo, auxilia o engenheiro de software a avaliar tanto a completitude dos interesses que foram modularizados em casos de uso aspectuais, quanto informações sobre a separação de interesses, tamanho e acoplamento dos casos de uso aspectuais.

Um ponto importante a ser notado na adaptação da metodologia é que existem dois tipos (duas grandezas) de métricas, uma para medir a completitude de um determinado interesse e outra para medir questões relacionadas a separação de interesse, tamanho e acoplamento dos casos de uso aspectuais. Por esse motivo optou-se por não colocar as métricas relacionadas com a medição da separação de interesses para atribuir os escores aos atributos de qualidade. Optou-se então em criar uma fase adicional que tratasse dessa medição.

A fase de Medição da Separação de Interesses, Tamanho e Acoplamento dos Casos de Uso Aspectuais, pode ser utilizada separadamente das outras, quando a intenção for saber informações no contexto do desenvolvimento orientado a aspectos. Porém o objetivo de utiliza-la juntamente com a metodologia é elaborar uma base de conhecimento para cada interesse medido.

Um problema que deverá ser encontrado, é que existem diversas abordagens para gerar documentos de requisitos orientados a aspectos, a metodologia apresentada neste artigo auxilia a verificação da qualidade em documentos de requisitos direcionados a casos de uso e orientados a aspectos. Portanto, como descrito na seção anterior novos estudos de

caso deverão ser realizados com outras metodologias de medição de qualidade na fase de requisitos, para que deste modo seja possível elaborar uma metodologia para avaliar a qualidade de qualquer documento de requisitos orientado a aspectos independente do domínio e da plataforma.

Como trabalhos futuros os dados da base de conhecimento adquiridos com a aplicação da fase de Medição da Separação de Interesses, Tamanho e Acoplamento dos Casos de Uso Aspectuais deverão ser adicionados juntamente ao catalogo de árvores de interesses e sub-interesses, para que assim possa fornecer um fator base para que o engenheiro de software tenha informações como por exemplo: quantos aspectos seriam uma boa quantidade para se modularizar um determinado interesse. Outro trabalho futuro será a obtenção de novas árvores de Interesse e Sub-Interesses para assim tornar o catalogo mais completo.

Referências

1. Aksit, M.; Tekinerdogan, B. and Bergmans, L. "The Six Concerns for Separation of Concerns", in Proceedings of ECOOP 2001 Workshop on Advanced Separation of Concerns, Budapest, Hungary, June, 2001.
2. Alves, C. F., Castro, J. F. B. CRE: A Systematic Method for COTS Componentes Selection. SBES 2002, Rio de Janeiro. Outubro, 2002.
3. Alves, C. F. Seleção de Produtos de Software Utilizando Uma Abordagem Baseada em Engenharia de Requisitos. Dissertação (mestrado em Ciência da Computação) – Programa de Pós-Graduação em Ciência da Computação, UFPE, Recife – Pernambuco, maio 2002.
4. Brito, I. and Moreira, A. “Towards a Composition Process for Aspect-Oriented Requirements”. Workshop on Early Aspects: Aspect-Oriented Requirements Engineering and Architecture Design, March 17 - Boston, USA, 2003.
5. Camargo, V.;Ramos, R. A.; Penteado,R.; Masiero, P.C; - “Projeto Baseado em Aspectos do Padrão Camada de Persistência” - SBES’2003 - Manaus/Amazonas- . Outubro, 2003.
6. Chung, L; Nixon, B.; Yu, E. and Mylopoulos, J. “Non-Functional Requirements in Software Engineering”. Boston Kluwer Academic Publishers, 2000.
7. Dijkstra, E. “A Discipline of Programming”. Prentice-Hall, 1976.
8. Durán, A.; Cortés, A. R.; Corchuelo, R.; Toro, M. “Supporting Requirements Verification Using XSLT ”.Requirements Engineering Conference - RE 2002 – Essen, Alemanha, Setembro 2002.
9. Elrad, T., Filman, R. and Bader, A. “Aspect-Oriented Programming: Introduction”, Communications of the ACM, v.44 n.10, p.29-32, Oct. 2001.
10. Elrad, T.; Aksit, M.; Kiczales, G.; Lieberherr, K. and Ossher, H. “Discussing Aspects of AOP”. Communications of the ACM, 44(10):33–38, October 2001.
11. Ghezzi, C.; Jazayeri, M. and Mandrioli, D. “Fundamentals of Software Engineering”. Prentice Hall, 1991.
12. ISO 9126, “Tecnologia da Informação – Qualidade de Produto – Parte 1: Modelo de Qualidade”, International Standard ISO/IEC 9126, International Standard Organization, Junho, 2001.
13. Jacobson, I; Chriterson, M; Jonsson, P. and Overgaard, G. "Object-Oriented Software Engineering: A Use Case Driven Approach". Addison Wesley, 1992.
14. Jacobson, I.; Booch, G.; and Rumbaugh, J. “The Unified Software Development Process”, Addison-Wesley, 1999.
15. Kiczales, G.; Lamping, J.; Mendhekar, A. RG: A Case-Study for Aspect-Oriented Programming. In: SPL97. Xerox Palo Alto Research Center, Technical Report, 1997.
16. Kiczales, G.; Hilsdale, E.; Hugunin, J.; Kersten, M.; Palm, J. and Griswold, W. “An Overview of AspectJ”. In J. L. Knudsen, editor, 15th European Conference on Object-Oriented Programming, volume 2072 of LNCS, pages 327–353, Berlin, Heidelberg, and New York, Springer Verlag, 2001.

17. Kitchenham, B.; Pickard, L. and Pfleeger, S. “Case Studies for Method and Tool Evaluation” . IEEE, 1994.
18. Moreira, A.; Araújo, J. and Brito, I. “Crosscutting Quality Attributes for Requirements Engineering”, 14th International Conference on Software Engineering and Knowledge Engineering (SEKE 2002), ACM Press, Italy, July, 2002.
19. Olsina, L.;Lafuente, G. J.; Rossi, G. E-commerce Site Evaluation: a Case Study, LNCS 1875 of Springer, 1st International Conference on Eletronic Commerce and Web Technologies, EC-Web 2000, London, UK, pp. 239-252. 2000
20. Pressman, R. Engenharia de Software. Makron Books, 5ª edição, 2002.
21. Ramos, R. A.; Penteado, R.; Masiero, P. C.; - “Um Processo de Reestruturação de Código Baseado em Aspectos” - SBES’2004 - Brasília/DF. Outubro, 2004.
22. Rashid, A.; Sawyer, P.; Moreira, A. and Araújo, J. “Early Aspects. A Model for Aspect-Oriented Requirements Engineering”. IEEE Joint Conference on Requirements Engineering, Essen, Germany, September, 2002.
23. Rashid, A. Moreira, A. and Araujo, J. “Modularisation and Composition of Aspectual Requirements”. 2nd International Conference on Aspect-Oriented Software Development, ACM, pp. 11-20, 2003.
24. Reis, T., P., C. “Uma Metodologia para Medição da Qualidade de Aplicações Web na Fase de Requisitos”. Dissertação (mestrado em Ciência da Computação) – Programa de Pós-Graduação em Ciência da Computação, UFPE, Recife – Pernambuco, 2004, 163f.
25. Sant’Anna, C.; Garcia, A.; Chavez, C.; Lucena C.; Staa A. On Reuse and Maintenance of Aspect-Oriented Software: An Assessment Framework. In: SBES’2003, Manaus/Amazonas. Outubro, 2003.
26. Sant’Anna, C.; Garcia, A.; Chavez, C.; Lucena C.; Staa A. “Design Patterns as Aspects: A Quantitative Assessment”. In: SBES’2004, Brasilia - DF, 2004.
27. Sousa, G.; Silva, I. and Castro, J. "Adapting the NFR Framework to Aspect-Oriented Requirements Engineering". In: SBES’2003, Manaus/Amazonas, Outubro. 2003.
28. Sousa, G. “Uma Abordagem Direcionada a Casos de Uso para o Desenvolvimento de Software Orientado a Aspectos”. Dissertação (mestrado em Ciência da Computação) – Programa de Pós-Graduação em Ciência da Computação, UFPE, Recife – Pernambuco, maio 2004, 180f.
29. Travassos, G.; Gurov, D. and Amaral, E. “Introdução à Engenharia de Software Experimental”. Relatório Técnico, RT-ES-590/02, Rio de Janeiro, 2002.
30. W3C. Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation. Outubro 2000.