

## 8

### Considerações Finais

O DSOA é um paradigma recente que promete aumento da qualidade em sistemas de software, por meio de uma melhor separação de interesses. Entretanto, alguns atributos de qualidade importantes podem ser afetados negativamente pelo uso inadequado das abstrações existentes neste paradigma. O DSOA vem ganhando força tanto na academia quanto na indústria, mas, apesar de sua popularidade, pouca atenção tem sido dada a métodos de avaliação que mostrem que sistemas desenvolvidos neste paradigma satisfazem a certas propriedades desejáveis. A pesquisa atual nesta área tem se preocupado principalmente com a construção de linguagens de modelagem [46] e implementação [56] [66]. Apenas alguns estudos experimentais [8] [11] [27] [36] têm sido realizados no contexto de avaliação, mas estes são predominantemente qualitativos e específicos de um domínio de aplicação [36] ou interesse [44]. Neste sentido, este trabalho define uma abordagem sistemática para auxiliar engenheiros de software na avaliação da qualidade de seus artefatos. A abordagem apresentada neste documento é composta de três elementos principais: um método de avaliação, um conjunto de regras heurísticas e uma ferramenta de suporte.

#### 8.1.

##### Contribuições

A medição de propriedades do sistema (tais como acoplamento, coesão, tamanho e separação de interesses) tem se mostrado uma prática promissora para prever a qualidade nas fases de projeto e implementação do DSOA [21] [57]. Neste contexto, este trabalho apresenta as seguintes contribuições principais:

- Um método de avaliação de sistemas orientados a aspectos. O método é iterativo, organizado em etapas, apoiado por métricas e por regras heurísticas. Este método encontra-se publicado em [21].
- Um conjunto de regras heurísticas orientado a interesses para projeto e implementação. Tais regras são associadas aos resultados de

medições e oferecem informações semânticas sobre os interesses e componentes do sistema. Algumas regras propostas nesta dissertação foram publicadas em [21].

- Uma ferramenta implementada e documentada para dar suporte às atividades do método de avaliação e às regras. Um dos benefícios centrais da ferramenta é automatizar a geração de alertas de possíveis problemas relativos a interesses transversais, apoiando engenheiros de software na identificação de refatorações que possam ser necessárias. A ferramenta é um projeto *SourceForge* hospedado em [53].
- Cinco estudos experimentais envolvendo implementações OO e OA para avaliação do método, regras heurísticas e ferramenta. É importante ressaltar que estes estudos também contribuíram para o amadurecimento da abordagem de avaliação proposta. Alguns destes estudos encontram-se publicados em [8] [27]. Vale ainda dizer que o primeiro estudo foi convidado especial para publicação na edição inaugural de LNCS Transactions on Aspect-Oriented Software Development [28].

Nos estudos realizados, a abordagem baseada em regras tem se mostrado bastante útil para apontar problemas não triviais resultantes de uma análise equivocada das medições. Como resultado, as métricas, as regras e o método de avaliação têm sido utilizados por pesquisadores em universidades internacionais como Nélcio Cacho da Universidade de Lancaster (UK) e Hans-Arno Jacobsen da Universidade de Toronto (Canadá). Além disso, a ferramenta de medição e avaliação que suporta a abordagem vem ganhando rápida visibilidade entre pesquisadores de diferentes instituições. Três exemplos podem ser citados: (i) a ferramenta foi utilizada no último estudo experimental deste trabalho, realizado em parceria com o aluno de mestrado Fernando Castor na Universidade Estadual de Campinas (UNICAMP); (ii) o módulo de medição da ferramenta está sendo estendido para comportar novas métricas pelo aluno de graduação Gary Thewlis, orientado pelo professor Alessandro Garcia, da Universidade de Lancaster – Inglaterra; e (iii) a ferramenta vem sendo estendida para suportar múltiplas linguagens de programação OA em parceria com o aluno de mestrado Thiago Bartolomei da Universidade de Ciências Aplicadas de Kiel – Alemanha. Esta

última extensão da ferramenta será feita em dois estágios: primeiro para suportar CaesarJ [51], e depois a generalização para outras linguagens utilizando mecanismos de *plugins*. Todos estes trabalhos colaborativos que vêm sendo realizados, envolvendo diversas pessoas e instituições, só vêm contribuir para o crescimento da abordagem e da ferramenta.

## **8.2. Trabalhos Futuros**

O trabalho apresentado nesta dissertação pode ser continuado com a realização das atividades apresentadas a seguir.

### **Método de Avaliação, Métricas e Regras Heurísticas**

- Definição de novas métricas e regras heurísticas para extensão do método de avaliação. As novas regras podem combinar uma ou mais das métricas existentes, ou mesmo utilizar novas métricas incorporadas ao método, para identificação de outras características importantes de interesses (ou componentes) na avaliação de qualidade do software. De fato, através da nossa experiência empírica, percebemos que o método deveria também prover suporte explícito a outras dimensões complementares àquelas que já são suportadas, tais como complexidade das interfaces aspectuais, e interação entre aspectos.
- Extensão do método proposto de tal forma a associar as regras definidas pelo método com possíveis sugestões de refatorações, melhorando a automatização dos processos de projeto da qualidade e manutenções periódicas.
- Definição de um conjunto de refatorações orientadas a aspectos para compor o método, ou selecioná-las dentre as existentes na literatura [35] [38] [52]. Estas refatorações devem ser utilizadas no método de avaliação como possíveis soluções para resolver problemas de projeto e implementação apontados pelas regras heurísticas.
- Propor métricas e regras para guiar o engenheiro de software nas decisões que ocorrem antes da fase de implementação, ou seja, nos

níveis preliminares de desenvolvimento. Uma abordagem à priori é importante porque ela pode evitar que problemas de interesses transversais ocorram durante a implementação. E, portanto, evitar refatorações do sistema que geralmente são custosas.

### Ferramenta AJATO

- Extensão da ferramenta para tornar possível medir e avaliar sistemas implementados em outras linguagens de programação. Este trabalho já se encontra em andamento por uma iniciativa de software livre, hospedado em [53]. A idéia é extensão da ferramenta por adição de *plugins* que descrevem as novas linguagens a serem incorporadas. Cada *plugin* deve definir atributos relevantes da linguagem como a sua gramática e seu analisador léxico.
- Extensão da ferramenta para avaliar artefatos no nível de projeto detalhado, além de artefatos de código. Os artefatos de entrada em uma avaliação no nível de projeto devem constar diagramas estruturais e dinâmicos UML [5] como, por exemplo, diagramas de classes e diagramas de sequência.
- Desenvolvimento de um novo módulo na ferramenta para permitir aplicação de refatorações OA em programas. Esta atividade é muito importante visto que o método de avaliação identifica problemas que podem ser resolvidos por refatorações OA, entretanto, a ferramenta ainda não suporta esta atividade. É importante ressaltar que o ambiente MetaJ [55] permite análise, transformação e geração de código, portanto, as refatorações podem ser implementadas utilizando os recursos deste ambiente.

### Estudos Experimentais

- Realização de um estudo mais aprofundado em relação ao quinto estudo de caso, Telestrada (Subseção 7.1.5). A avaliação neste estudo foi baseada exclusivamente em métricas, ou seja, o método de avaliação e as regras heurísticas não foram utilizados. Desta

forma, torna-se natural uma avaliação que aplique todas as etapas da abordagem proposta.

- Realização de novos estudos experimentais para cobrir outras variáveis da abordagem como, por exemplo, sistemas implementados em diferentes linguagens. Os cinco estudos realizados neste trabalho se limitaram a estudos comparativos entre implementações Java e AspectJ e outros estudos tornam-se importantes para maior amadurecimento da abordagem. Entretanto, é importante ressaltar que (i) os sistemas avaliados possuem variados graus de complexidade e cobrem diferentes domínios, e (ii) as linguagens utilizadas são as mais difundidas e realizam as principais abstrações dos paradigmas OO e AO, respectivamente.