9
Conclusões

Este Capítulo descreve as conclusões gerais desta tese, resume suas contribuições e propõe direções para trabalhos futuros.

Esta tese tratou de problemas atuais relacionados ao design e à modelagem orientados a aspectos.

A análise de algumas abordagens que dão suporte à programação orientada a aspectos resultou em um framework conceitual unificador para a POA, a *teoria dos aspectos*. Essa teoria de aspectos foi descrita informalmente e foi organizada em quatro modelos: o modelo de componentes, o modelo de pontos de combinação, o modelo de processo de combinação e o modelo principal.

Para validar as idéias da teoria, foi proposto um metamodelo para a UML, o metamodelo `aSide`. A arquitetura do metamodelo `aSide` foi diretamente derivada do modelo de aspectos. O metamodelo `aSide` especifica a semântica de `aSideML`, uma linguagem de modelagem para o design orientado a aspectos.

A linguagem `aSideML` provê semântica, notação e regras que permitem que o projetista construa modelos cujo foco são os principais conceitos, mecanismos e propriedades de sistemas orientados a aspectos, nos quais os aspectos e *crosscutting* são explicitamente tratados como cidadãos de primeira classe.

Para demonstrar que a notação de `aSideML` facilita a visualização, a compreensão e a comunicação de modelos de design orientados a aspectos, revisitamos o SMA Portalware, descrevendo seus modelos estáticos e dinâmicos orientados a aspectos com `aSideML`.

9.1
Contribuições

As contribuições deste trabalho foram descritas no Capítulo 1, Seção 1.4. Aqui, reforçamos essas contribuições e tecemos comentários adi-

cionais.

São contribuições desta tese:

1. A *teoria de aspectos* (o *modelo de aspectos*), um framework conceitual que representa uma descrição informal e preliminar das propriedades e dos conceitos fundamentais do desenvolvimento de software orientado a aspectos. A teoria de aspectos pode ser usada para avaliar o que é orientado a aspectos e o que não é e para ajudar no projeto de linguagens orientadas a aspectos. Essa teoria foi apresentada no Capítulo 3.

2. `aSideML`, uma linguagem de modelagem para especificar e comunicar designs orientados a aspectos nos quais aspectos e *crosscutting* são explicitamente tratados como cidadãos de primeira classe. A linguagem `aSideML` foi apresentada no Capítulo 5.

   A linguagem `aSideML` tem sido usada para a modelagem de soluções orientadas a aspectos em projetos ligados ao Grupo SoC+Agents [126], na PUC-Rio.

3. O metamodelo `aSide`, um modelo lógico que define a semântica dos modelos comportamentais e estruturais suportados pela `aSideML`. Ele é definido como um núcleo fundamental que oferece suporte a aspectos e *crosscutting* e que pode ser estendido para oferecer suporte a modelos específicos de algumas linguagens. O metamodelo `aSide` foi apresentado no Capítulo 6.

4. Um conjunto inicial de princípios e diretrizes que devem ser usados na modelagem orientada a aspectos. A linguagem `aSideML` foi usada para ilustrar as diretrizes apresentadas no Capítulo 8.

5. O uso de ferramentas e linguagens padrão e atuais. Construímos sobre tecnologias aceitas pela indústria e definimos:

   - um modelo de aspectos no nível de projeto que usa o Modelo de Objetos de UML.
   - `aSideML`, uma linguagem de modelagem orientada a aspectos com base em uma extensão de UML.
   - um mapeamento preliminar de `aSideML` para AspectJ e Hyper/J.

Essas contribuições foram parcialmente descritas em alguns trabalhos e relatórios técnicos [27, 26, 22, 23, 24, 25] e foram ilustradas através de

um estudo de caso apresentado no Capítulo 7 e exemplos apresentados no Capítulo 8.

Ademais, a teoria de aspectos tem se mostrado bastante útil como ferramenta de ensino e base para discussão sobre conceitos e propriedades da programação orientada a aspectos e abordagens relacionadas.

9.2 Trabalhos Futuros

Este trabalho pode ser elaborado de várias maneiras.

Modelagem orientada a aspectos e `aSideML`

Precisamos elaborar ainda mais sobre diversos problemas relacionados à expressividade da linguagem `aSideML`, em especial os aspectos parametrizados, a evolução de aspectos, a herança de aspectos e o suporte à quantificação. Esses problemas foram discutidos na Seção 5.7.

Aspectos e Ferramentas

A modelagem com `aSideML` precisa ser viabilizada na prática através do desenvolvimento de ferramentas CASE.

Ferramentas, em especial, são necessárias para realizar o processo de combinação, isto é, para gerar (automaticamente) os relacionamentos de *crosscutting* que usam *wildcards* e para produzir (automaticamente) os elementos combinados finais.

Uma vez disponíveis essas ferramentas, os projetistas poderão aproveitar os benefícios do modelo de design orientado a aspectos e "desenhar" explicitamente as instruções dos processos de combinação.

Aspectos e Model-Driven Architectures

A nova versão de UML 2.0 oferece suporte a `InfraStructure Library`, uma biblioteca que inclui um novo pacote `Core`. O pacote `Core` é um metamodelo completo desenvolvido especialmente para melhorar a reusabilidade quando outros metamodelos no mesmo metanível importam ou especializam suas metaclasses especificadas. A intenção é que metamodelos de UML e outros metamodelos de MDA reutilizem partes desse pacote.

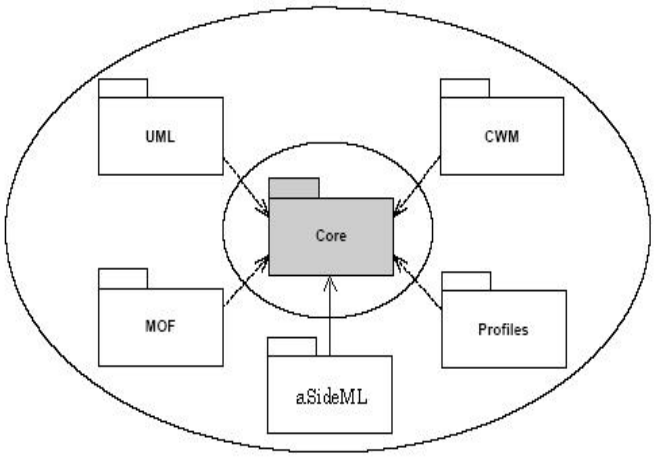

Figura 9.1: Conformidade com MOF.

Isso é ilustrado na Figura 9.1, onde é mostrado como `aSideML`, UML, CWM e MOF dependem de um núcleo comum. O metamodelo `aSide` pode ser adaptado com facilidade a fim de aproveitar a semântica e a sintaxe abstrata já definidas no pacote `Core`. Ademais, ele importa e adota o metamodelo de UML como seu modelo de componentes e oferece suporte a uma extensão conservativa (Seção 6.1.2).

Aspectos e Agentes

Uma possível área de investigação é o uso do *modelo de aspectos* associado ao *modelo de agentes*, combinando os novos mecanismos e abstrações de aspecto [27] com os novos mecanismos e abstrações de agente propostos em [120].

Ao considerar um *agente de software* como componente base, possíveis pontos de combinação de seu modelo seriam "entrada em organização", "saída de organização", "execução de plano", "realização de objetivo", etc.

Aspectos e Métricas

O modelo de aspectos pode ser usado como um framework conceitual para a definição de uma ferramenta CASE que ofereça suporte à modelagem orientada a aspectos com `aSideML` e um conjunto de métricas para o DSOA [118]. O modelo usado atualmente no framework proposto em [118] é o modelo de implementação da linguagem AspectJ.

Também são necessárias outras investigações experimentais para avaliar a forma como a POA interfere nos princípios de orientação a objetos existentes, conforme discussão apresentada na Seção 8.1.

Aspectos e Arquitetura de Software

Há alguma semelhança entre aspectos e *conectores de software* como são descritos nas linguagens de descrição arquitetural. Os *conectores de software* são o local das relações entre os componentes arquiteturais; eles são blocos básicos usados para modelar as interações entre os componentes e as regras que governam essas interações [119]. No entanto, modelos de conectores não oferecem suporte explícito à modelagem de *concerns* transversais. Estudos nessa direção ainda são necessários a fim de investigar as possíveis relações entre aspectos e conectores.