MDA 建模的 AOP 扩展策略及其比较 *

余金山
（华侨大学 计算机学院，福建 泉州 362011）

摘 要：对于 MDA 横切于核心业务逻辑的关注点对封装的破坏的问题，本文给出把 AOP 引入到 MDA 的扩展策略和主要方法，并对不同的扩展策略进行了比较。

关键词：模型驱动架构；面向方面；扩展；元模型；统一建模语言
中图分类号：TP311.5 文献标识码：A 文章编号：1674-7720(2011)24-0001-04

Strategies and comparison of MDA modeling extension for AOP

Yu Jinshan
(College of Computer Science and Technology, Huaqiao University, Quanzhou 362011, China)

Abstract: MDA is unable to effectively deal with the conerns that crosscutting the core business logic and causing the destruction of encapsulation. In the article, several extension strategies and main approaches of how to introduce AOP to MDA are discussed, and different extension strategies are compared.

Key words: model-driven architecture(MDA); aspect-oriented; extension,; metamodel; unified modeling language(UML)

MDA 是一种以模型为中心的软件开发新方法[1,2]。但是，MDA 仍然不能有效地处理横切“关注点”问题。

近来 MDA 和 AOSD 两者的融合和相互支持已成为具有重要的理论意义和实用价值的研究课题[1]。其中的主要研究方向之一是：采用主流开发框架 MDA，在 MDA 的建模过程中引入 AOP 的概念和思想，建立通用的具有面向方面特性的 PIM。

如何把面向方面的概念和思想引入到 MDA 中，可以采取多种不同的扩展策略。

1 MDA 引入 AOP 的扩展策略

在 MDA 的建模过程中引入面向方面的思想和概念，其中最重要的是如何表示面向方面 AOP 的概念和思想以及表达方式。

下面给出两种有代表性的扩展策略及其实现方法：基于 MOF 的扩展和基于 UML 2.0 Profile 的扩展，这是当前的主流。

1.1 基于 MOF 的扩展策略

在 MDA 中，MOF 位于 MOF 四层模型的最高层。MOF 体系是开放的，因而可以对它添加新的描述 UML 的元类型，以扩展新的功能和应用。图 1 给出了这种扩展的示意。

但是，这种扩展策略要做的工作较多也较为困难，必须对 UML 的核心语义十分熟悉。目前所做的研究还很少。基于 MOF 的 AOP 扩展也称为“重量级”的扩展。

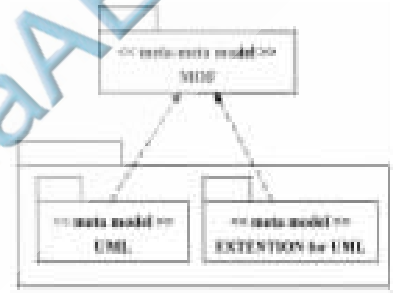

图 1 基于 MOF 的扩展策略

参考文献[2]比较全面地介绍了一种通用的支持 AOP 的 MOF 2.0 元模型。

MOF 扩展策略以 MDA 作为不同的 AOP 扩充方案之间的集成元素，自动或半自动地把在某开发阶段建立的模型转换成下一阶段的模型，直到最终实现为止。首先建立 AO 术语的被广泛接受的本体，基于这些本体的共同元素和它们之间的关系建立通用的支持 AOP 的 MOF 元模型。其扩展方法和过程可描述如下：

第一步是确定出各种不同的 AOP 扩充方案所共享的概念，并根据概念的目的把它们分组成包，以提高可理解性。然后，建立包与包之间的关系。由此，可以建立三个主要的包：(1)Entities 包，描述分解单元和它们之间的关系的实体；(2)JoinpointModel 包，包含允许注入行为对应用程序的执行进行干预的关注点的包；(3)CompositionRules 包，它有两个子包，SymmetricRules 描述对称规则，AsymmetricRules 描述不对称规则。这三个包可以从

* 基金项目：福建省自然科学基金资助(A0810013)

UML 2.0 元模型的内核和 CommomBehaviors 包的元素进行特化。

实体：UML2.0 BehaviouredClassifiers 的特化。

JoinpointModel：连接点是对应用程序的运行进行干预以执行一个方面行为(AspectJ 中的 advice)。所以，每个实体由一个可以隐式附上方面行为的 Hook 集组成。在 UML2.0 中，这些 Hooks 是基本单元的 Behaviors。

CompositionRules：SymmetricRules 用以通过若干实体(也称为子模块)的组合产生一个更大的实体。AsymmetricRules 用于说明任何在连接点(hooks)上应用 aspects。

这样建立的通用元模型可以精细化为更加具体的模型，例如 CBSD(Component-Based Software Development)与 AOSD 的集成模型 CAM、Theme/UML 模型。CAM 与 Theme/UML 模型之间还可以实现相互转换。

1.2 基于 UML 2.0 Profile 扩展策略

这种扩展策略的基础是：UML2.0 Profile 提供了很好的扩展机制。图 2 是其示意图。

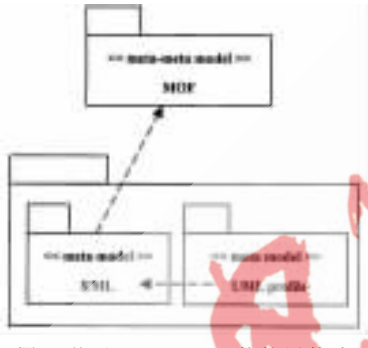

图 2 基于 UML Profile 的扩展策略

1.2.1 扩展方法和扩展机制

UML2.0 是建模语言的核心，其本身就是可扩展的。基于 UML Profile 的扩展策略实际上是一种基于 UML 2.0 的扩展。基于 UML 2.0 的扩展策略有两种方法：

(1) First-Class 扩展。它允许增加、修改具有相应功能的元类就能使得 UML 2.0 具有描述 AOP 特性的建模能力。这种扩展增加了 UML 2.0 语言的复杂性和应用建模的困难性，还涉及到 MOF 的修改，对已有的标准化造成了破坏，因而其缺点是明显的。

(2) Profiles 方式。它是一种通过建立面向方面 PIM 的 PIM-AOP Profile 的方式。Profiles 扩展无需改动已有的元模型，也不改变现有的语法和语义，而是通过 UML Profiles 的构造型、标记值和约束等扩展机制来扩展 UML 2.0 的建模能力。这种方法易于理解，不破坏现有标准，还可以充分利用现有的相关工具。

因此，通常选择后一种扩展方法。它利用了 UML 2.0 Profiles 提供的三种扩展机制：

(1) 构造型

该机制利用一个已存在的模型元素来定义一种新的模型元素。新模型元素是已有模型元素的子类，通过泛化继承已有的元模型类，并添加额外的语义来实现扩展。

(2) 标记值

标记值用于指定模型元素的性质，由标记字符串和值字符串组成，是对特性的显式定义。任何模型元素都可以应用标记值来增加新的语义。标记值也可用来提供附加信息以有效提高模型的质量。

(3) 约束

约束用于对模型元素提供语义条件或限制。它用大括弧内的字符串表达式来表示。约束可应用于一个或多个元素上，可以附加在依赖、注释等元素上以表示约束和关系。因此可以用来对模型元素进行语义上的限制。

1.2.2 Profiles 扩展方式实例

(1) 方面 aspcet 的表示。方面是 AOP 最主要的概念，在建模过程中应该把它作为一个独立的、封闭的并且可以拥有自定义的结构成员的实体来定义。Aspect 可以通过 UML Profiles 的构造型的扩展来实现。如图 3 所示。

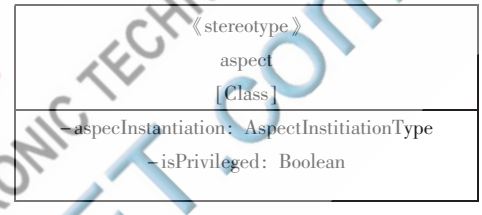

图 3 aspect 的构造型

在定义 Aspect 构造型时需要：(1)指出新定义的构造型是基于哪个元素的；(2)改变或增加新的语义。

标记 isPrivileged 用于决定 aspect 是否能访问被其横切的类的私有属性。

标记 aspectInstantiationType 用来决定 aspect 实例化的方式。它可定义基于连接点中对象实例化的、基于连接点中的控制流的和在整个应用程序中只创建一个 aspect 对象的等几种方式。

利用方面 aspect 既可以扩展出新的方面 aspect 或者抽象方面 abstractaspect，也可以通过实现接口方面 Iaspect 来构造方面 aspect。

(2) 切入点 pointcut 的表示。切入点、连接点的定义必须包含静态语义和动态行为语义两个方面。切入点、连接点的 AOP 动态行为语义是指干预应用的运行以执行方面 aspect 的行为。动态行为语义的描述和扩展通常选用在 AOP 中具有代表性的 AspectJ 中的 advice。

切入点的表述可通过 UML Profiles 的构造型的扩展得到构造型 pointcut。并可(也必须)为 pointcut 添加约束，使其只能用于构造型 aspect。

(3) 通知 advice、连接点 joinpoint。与切入点相关的通知 advice 和连接点 joinpoint 仍然可以通过 UML Profiles 的构造型的扩展来实现。

用于描述动态行为语义的 AspectJ advice 实际上就是 UML 2.0 的 Behaviors。对构造型 advice 可以且必须添

加约束,使其只能用于构造型 pointcut。并且可以实现通知的五种执行方式:

① before 通知:在连接点 joinpoint 执行之前执行 before advice。但不能阻止连接点前的执行;

② after 通知:在连接点 joinpoint 执行之后执行 after advice;

③ around 通知:包围一个连接点的通知,例如方法调用。Aroud 通知在方法调用前后完成自定义的行为,并负责选择继续执行连接点或通过自己的返回值或抛出异常来中断执行。

④ AfterThrowingAdvice 通知:在方法抛出异常时执行的通知。

⑤ AfterReturningAdvice 通知:在连接点正常完成后执行的通知。

连接点有两种类型:类连接点和方法连接点。

类连接点构造型 classjoinpoint 的扩展实现可以也需要添加约束,使其只能用于切入点 pointcut。并且可以实现通配符的条件比较。同样,方法连接点构造型 methodjoinpoint 的扩展实现也需要添加约束,使其只能用于切入点 pointcut。

方法连接点 methodjoinpoint 是专门针对方法的。方法所具有的修饰符、返回值、方法名、参数、参数个数以及异常等的表述,都可以在此基础上,将方法连接点 methodjoinpoint 分解为修饰符构造型 modifierJ、返回值构造型 returnJ、参数构造型 argsJ 以及异常构造型 exceptionJ。对 modifierJ、returnJ、argsJ 也都可以添加约束,使得它们只能用在方法连接点构造型 methodjoinpoint 和类连接点构造型 classjoinpoint 中。

此外,AOP 中对连接点的动态捕获(控制流程中类方法执行中涉及的连接点以及对类的字段进行处理、类初始化、对象初始化等涉及的所有连接点)和静态捕获(捕获在指定类或者方面中的程序体中的所有连接点以及满足一定条件的连接点),以及用于表示横切关注点引发的行为顺序等机制,都可以通过 UML 2.0 Profile 的扩展来获得。

2 扩展策略的比较和相关研究

相对来说,使用 UML 2.0 Profile 的最大好处是可以获得大多数 UML2.0 工具的支持,建模人员可以容易地使用 UML2.0 Profile 扩展机制在 MDA 中引入面向方面的建模[9,13]。从理论上来说,可以创建无限个特征文件 Profile。因而,只要遵循 UML 2.0 规范,任何的建模人员都可以创建满足自己需要的任何特征文件。通过 Profile 的扩展机制建立 UML2.0 PIM-AOP Profile 模型规范的方式具有较强的通用性,例如当 AOP 的语法增加或变动时,可以容易地在该 PIM 模型规范中实现。

但是,在进行软件开发时使用 UML2.0 Profile 扩展,必须首先利用面向方面技术将核心业务逻辑和横切关注点分离,然后建立实现系统核心关注点的功能 PIM 和横切关注点的方面 PIM,接下来还要根据选定的具体技术平台利用相关的平台变换规则分别将功能 PIM 和方面 PIM 变换为功能 PSM 和方面 PSM。最后还需利用特定方面平台的编织技术将这两部分编织在一起形成最终代码[1]。

与 UML Profile 的扩展策略相比较,MOF 扩展可以充分利用和表达元模型的语义,其抽象级更高,因而灵活性也更高。但它工作困难、现有的支持工具较少。

支持 AOP 的 MOF 元模型具有更强的表达能力,而且独立于任何的标记,同时还能为除 MDA 以外的其他建模标准(如 XMI、QVT 等)所使用。

支持 AOP 的 MOF 元模型的优点是:(1)有助于不同的 AOP 扩充方案建立元模型;(2)通过一个简单的元模型能够方便地确定不同的 AOP 扩充方案的异同;(3)能够实现模型之间的水平变换,即不同的 AOP 扩充方案之间的转换;(4)有利于转换工具的开发。

MOF 扩展可以遵循 MDA 方法,使用预定义的转换,从元模型生成目标模型所有可能的元素。另一方面,利用这种方法,可以生成多个目标模型,获取相同系统的不同实现方案。

此外,基于各种特定的 MOF 元模型建立的任何模型,以及任何使用经扩展的通用的支持 AOP 的 MOF 元模型构建的任何模型,都可以直接通过 XMI 标准序列化为 XML。

一些试图扩展 UML 元模型以支持 AOP 的相关工作有:

参考文献[3]根据 AspectJ 和 AspectC++中元素的语义,通过继承相应 UML 中元素的方法扩展了 UML 的元模型。类图中的切入点可以从方面中分离出来,这样可以清晰地表示在这个切入点处的多个横切关系;顺序图中新添加的通知发出焦点可以方便地表达行为横切。但所采用的方法是非形式化的,而且其实现和应用都较麻烦和困难。

参考文献[4]基于 AspectJ 语法语义和 XMI 扩展 UML 元模型,并实现了面向方面的重要概念。但该方法缺乏对基于面向方面的动态行为和特征的支持。

这些基于 UML 元模型的扩展都非常接近于诸如 AspectJ 的原始 AOSD 方法,且不支持更高级的功能,如对称组合。

其他的一些对基于 UML Profile 扩展的研究工作有:参考文献[5]给出了面向方面设计的通用 UML Profile 的构造。提出了用 UML 中的 collaboration template、Feature 和 Association 模型元素的扩展构造型。但是工作并没有最终完成,而且寄希望于通过不同的方案来完成。此外,它仍然不支持诸如对称组合规则等高级概念,其实用性也有待于在不同的方案中进行检验。参考文献[6]提出了用于 AspectJ 的设计标记。参考文献[7]提出一个原始

的 AspectJ profile，其中，advice 和 pointcut 是基于 operation 的构造型，aspect 与核心业务模块之间的关系表达为依赖关系。参考文献[6]、[7]给出的扩展策略的共同缺点是过分依赖于特定的技术。

还有一些略有不同的扩展策略：(1)基于 UML profile 的扩展，同时又可通过 MOF 扩展引入新的元模型的方法；(2)在 UML 模型中引入构造型 aspect；(3)不需要扩展 UML，利用状态图和类图为不同的关注点进行建模。但这些方法均不完善且存在较多的缺点[8]。

以模型为中心的软件开发新方法 MDA 有其独特的绝对优势，面向方面技术可以对 MDA 进行补充，以解决 MDA 难以处理的横切关注点的问题。将面向方面的 MDA 用于复杂软件系统的开发可以大大提高软件的可重用性和可维护性，降低开发成本。将这两种重要的软件开发方法结合起来有望成为未来软件开发的发展趋势。但是如何将两者很好地结合起来还有许多问题需要进一步深入研究和解决。

参考文献

[1] 刘敬勇，张立臣，陈成.基于面向方面 MDA 的软件开发方法[J]. 计算机工程与设计，2009，30(17)：4077-4080.

[2] FUENTES L，SANCHEZ P. A generic MOF metamodel for aspect-oriented modeling [C]. Proceedings of the Fourth Workshop on Model-Based Development of Computer-Based Systems and Third International Workshop on Model-Based Methodologies for Pervasive and Embedded Software (MBD/MOMPES'06)，Potsdam，IEEE Computer Society，2006：113-124.

[3] 郭东亮，张立臣.基于扩展 UML 的面向方面的建模 [J]. 计算机工程，2006，32(19)：100-102.

[4] 吴刚，郝克刚，葛玮.基于 UML 的面向方面建模研究[J]. 计算机应用与软件，2007，24(11)：95-97.

[5] ALDAWUD O，ELRAD T，BADER A. Uml profile for aspect oriented software development[C]. In Proc.of 3rd Int. Workshop on AOM，AOSD 2003，Boston，USA，2003(3).

[6] STEIN D，HANENBERG S，UNLAND R. A UML-based aspect-oriented design notation for AspectJ[C]. In Proc. of AOSD'02，Enschede，The Netherlands，2002(4)：106-112.

[7] STEIN D，HANENBERG S，UNLAND R.Designing aspect-oriented crosscutting in UML[C].In Proceedings of the AOM with UML workshop at AOSD 2002.

[8] YAN H，KNIESEL G，CREMERS A. A metamodel and modeling notation for AspectJ[C]. In Proceedings of the AOM workshop at AOSD 2004.

(收稿日期：2011-09-16)

作者简介：

余金山，男，1952 年生，教授，主要研究方向：软件工程，网络计算和人工智能应用等。