一种构件平台上支持面向方面编程的方法[1]

李洪涛[1] 郑任持[1] 陈瑢[2] 殷人昆[3]
（1、清华大学深圳研究生院软件工程中心 北京 100084；2、清华大学信息技术研究院操作系统与中间件中心 北京 100084；3、清华大学计算机科学与技术系 北京 100084）

摘要：面向方面编程与构件技术的结合是开发更加可复用、可扩展的软件的新方法。不同于传统的通过修改系统架构以及引入新的编程语言实现二者的结合，本文提出一种将软件划分为构件与方面，利用 XML 语言描述构件与方面的编织关系，结合构件容器技术与面向方面编程的方法实现构件与方面的运行时编织的方法。新的方法既可以简单方便地描述方面，又有效地扩展了现有系统，原有构件不需要任何改动，就可以直接实现同方面的编织，同时确保比较高的运行效率。同时本文描述了该方法在 CAR 上的软件工程实践。

关键词：构件技术；CAR；面向方面编程
中图分类号: TP311.5

An Approach to Support AOP on Component Platform

LI Hong-tao[1] ZHENG Ren-chi[1] CHEN Rong[2] YIN Ren-kun[3]

(1.Software Engineering Center, Graduate School at Shenzhen, Tsinghua University, Beijing100084, China; 2.Koretide Co., Shanghai201203, China; 3.Department of Computer Science and Technology, Tsinghua University, Beijing100084)

Abstract: Aspect-oriented component engineering (AOCE) is a new methodology for component-based development. Different from traditional way to modify the system architecture or introduce new language, in this paper we introduce our approach to support AOP on component platform. In our approach applications are composed of components and aspects and composed at runtime according to the composition information descriped by XML. We introduce a composition mechanism of composing components and aspects by merging the component container technology and AOP approaches. In our approach, we can simply describe the aspects, weave the aspects into components without any change of components to extend the system architectures, and have little holdback of effiency. Also the approach 's implementation on CAR is described.

Key words: component technique， CAR， AOP

[1]本文的研究工作得到国家高技术研究发展计划（863 计划）项目的支持（项目编号 2001AA113400，2003AA1Z2090）。

1. 引言

基于构件的软件开发(CBSD，Component Based Software Development)，以可复用软件构件为组装模块，以软件架构为组装蓝图，支持组装式软件的复用，能有效的提高软件的生产效率与质量。CBSD 的复杂性主要集中如何将系统划分为不同功能的可组装的构件。在实际的开发中，需要将系统依据功能划分为不同的模块并将它们包装成可复用的构件。然而实际系统中，许多属性并不能很好的划分到这些依据功能划分的构件中[1] 。

面向方面编程(AOP, Aspect Oriented Programming)[2]，作为当今世界上方兴未艾的一门编程方法，在面向对象编程的基础上引入方面的概念，实现关注点的分离，改善系统逻辑，使得系统可以更好地模块化。

面向方面编程与构件技术结合，作为一种新的软件开发途径得到了广泛的关注[3],[4]。在系统的开发中将构件作为服务的提供者与使用者，描述系统的功能划分.将方面作为系统的横切(crosscut)，描述不能很好的划归到功能构件的系统属性，如用户界面、安全、分布性等。面向方面编程与构件技术的结合，提供了分离关注(Separation of Concerns)，降低了构件之间的依赖性，使得开发出的系统具有更大的鲁棒性、扩展性与重用性.

当前已经提出了几种支持面向方面编程与构件技术结合的方法，如 JBoss/AOP[5],JAsCo[6],DAOP[7].但这些方法或者引入新的编程语言，或者引入新的框架结构，并且实现都比较复杂.

CAR(Component Assembly Runtime)[8][9]是由科泰世纪公司开发的，具有自主知识产权的新一代的构件系统，其主要的目的是从操作系统层引入构件的概念，所有的服务由构件来提供，实现软件的目标代码级的复用，为网络编程和 Web 服务提供强大的支持。

在 CAR 的基础上，本文提出并实现了一种构件平台上支持面向方面编程的方法，实现面向方面编程与构件技术的结合。同现有的技术相比，方面的定义与实现简单，不需要引入新的编程语言；支持现有构件的使用，不需要对现有构件做出修改；构件与方面的组合关系通过 XML 语言描述，通过容器实现运行时组合，具有比较高的运行效率。

在本文以下章节中的，将首先介绍该方法，然后介绍该方法在 CAR 构件平台上实现。

2. 构件平台上支持面向方面编程的方法

在我们的方法中，软件被分解为构件与方面，构件描述系统的功能划分，方面描述不能划分到功能构件的系统属性。构件与方面作为组成软件的独立实体，运行时动态编织（weave），实现系统功能.(参考图 1)

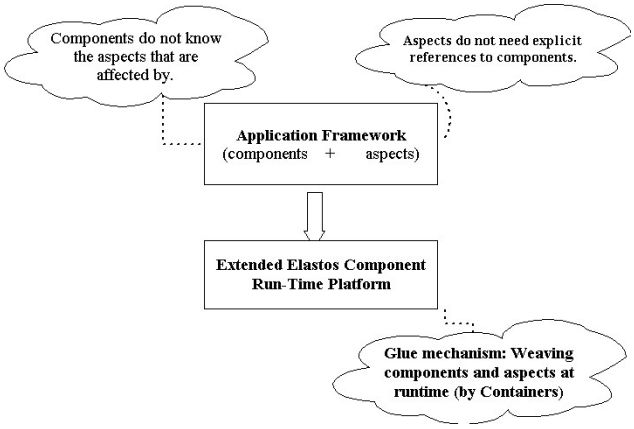

图 1 软件分解为构件与方面

在我们的方法中首先定义方面，然后描述构件与方面的编织（组合）关系。我们定义了方面与构件的编织规则，并利用 XML 语言具体描述构件与方面的编织关系。最后结合构件容器技术与面向方面编程的方法实现了一种构件与方面的依据编织关系描述进行编织的机制。

2.1 方面与方面构件

面向方面编程中最重要的概念是方面。在我们的方法中，构件作为服务的提供者与使用者，描述系统的功能划分，方面描述不能很好的划分到功能构件的系统属性，如安全、分布性等。

在我们的方法中方面的功能是由一类特殊的构件－－方面构件的接口方法来实现的。方面构件作为系统中的特殊构件，它是方面的二进制载体，它的对象中的接口方法实现方面的功能。一个方面构件中可以包含多个方面。方面通过方面构件 ID、类名、接口名、方法名、参数列表来描述。即一个方面的描述如下：

<AspectComponentID><ClassName><InterfaceName><MethodName>(<ParamList>)

通过将方面作为实现横切功能的方面构件的接口方法，可以通过现有的构件技术开发方面，避免了引入新的编程语言或对现有的构件技术做出修改。

2.2 编织规则

面向方面编程中，为了将独立实现的一般关注点与横切关注点组合起来形成最终系统，需要建立一种描述组合不同的实现单元以形成最终系统的规则的语言。这种指定编织(weave)规则的语言可以是实现语言的扩展，如 AspectJ[10]，也可以是一种独立的描述语言。

为了实现构件与方面的编织，我们参照 AspectJ 中对编织规则的描述，定义了构件与方面的编织规

则。编织规则通过切入点(point cuts)，连接点(join points)，通知(advice)来说明。

在 AspectJ 中，方面被定义为跨过多个模块的关注点，连接点是程序流中指定的一点，切点是特定的连接点的集合，一个通知是当一个连接点到达时执行方面代码的规则。AspectJ 作为支持面向方面编程的 Java 扩展，可以使用方面横切程序流中的任意一点。

在构件技术中，构件只能暴露出接口方法，而不应暴露内部的实现[8]。为此，我们将连接点定义为构件中对象的接口中的方法，通过构件 ID、类名、接口名、方法名、参数列表描述，即：

<ComponentID><ClassName><InterfaceName><MethodName>(<ParamList>)

我们将切点定义特定的连接点的集合，切点中方法名支持通配符的使用。例如在方法名中使用通配符可以指定一类符合匹配规则的接口方法。

通知（Advice）描述方面对切点的行为。我们定义了三种类型的通知：

Before:：在到达一个连接点但是在执行这个连接点内容之前执行方面.

.After：在一个连接点执行完成之后执行方面.

.Around:：在一个连接点执行前与执行完成之后均执行方面.

通过连接点、切点、通知、方面我们能有效的描述方面对构件的横切。

2.3 编织描述

编织规则定义了方面与构件的组合规则。方面依据编织规则对构件进行横切。我们基于编织规则，通过 XML 语言具体描述一个方面对一个构件的横切关系。基于 XML 语言使得规则的描述清晰，易于扩展与实现分析工具。

在 XML 中我们定义描述方面，构件，规则的标签。标签的名称与编织规则中的定义一致。

例如,给定一个构件 ID 为 *252C43D4-A20F-43cd-AEF1-BAA4C95A48B3* 的 Hello 构件，其 CHello 对象的 IHello 接口的 SayHello 方法作为连接点:

< *252C43D4-A20F-43cd-AEF1-BAA4C95A48B3*><*CHello*>< *IHello*>< *SayHello*>，

同时给定一个构件 ID 为 *98e1ab23-8dbe-4cde-b8e9-6e6aefb0b191* 的 Greeting 方面构件,其 CGgreeting 对象的 IGreeting 接口的 AfterGreeting 方法作为方面:

< *98e1ab23-8dbe-4cde-b8e9-6e6aefb0b191*><*CGreeting*>< *IGreeting*>< *AfterGreeting*>，

需要在 SayHello 方法每次执行之后执行该方面，则编织关系的 XML 描述为：

```
<Weaving>
  <Rule>
    <JoinPoint>
      <ComponentID>
        252C43D4-A20F-43cd-AEF1-BAA4C95A48B3
      </ ComponentID >
      <ClassName>
        CHello
      </ClassName>
      <InterfaceName>
        IHello
      </InterfaceName>
      <MethodName>
        SayHello
      </MethodName>
    </JoinPoint>
    <Aspect >
      <AspectComponentID>
        98e1ab23-8dbe-4cde-b8e9-6e6aefb0b191
      </ AspectComponentID >
      <ClassName>
        CGreeting
      </ClassName>
      <InterfaceName>
        IGreeting
      </ InterfaceName>
      <MethodName>
        AfterGreeting
      </MethodName>
    </Aspect>
    <Advice>
      After
    </Advice>
  </Rule>
</Weaving>
```

需要说明的是这个例子中没有涉及参数的处理，如果方面需要处理连接点的输入输出参数，需要在连接点与方面中对参数做出详细的描述。

2.4 构件与方面的动态编织

为了实现方面对构件的水平切割，我们需要提供一种机制将方面与构件在运行时按 XML 语言中描述的编织规则进行编织。

在我们的方法中，设计了一种编织机制，该机制根据编织描述，生成容器（Container），在容器中实现方面与构件组合，从而实现了方面与构件的动态编织。相比于其它的编织机制，如 DAOP[7]的消息机制，基于容器的编织机制避免对构件的修改，支持对现有构件的使用，并且具有比较高的运行效率[9]。

2.4.1 构件容器技术

以 Cobra 和 COM+为代表的容器技术，可以在不修改应用程序的基础实现其它的服务支持，例如分布式服务支持等（参考图 2）。 以 CobraComponent Modol (CCM)为例，程序员只需要关注自己的构件实现，通过使用 CCM Container 来透明的提供其它服务[11]。

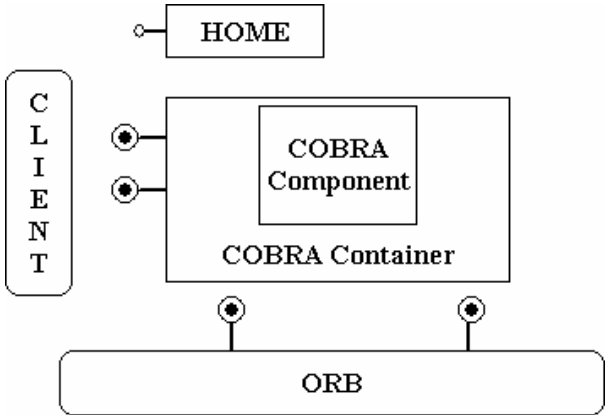

图 2 COBRA 中的容器模型

容器暴露与构件相同的接口，所有对该构件接口方法的调用事实上都是对容器相应接口方法的调用。容器中相应接口方法负责转发对构件接口方法的调用并实现横切[12],[13]。

2.4.2 容器生成技术

在实现容器生成时，我们的方法中提供了两种方法。第一种是运行前生成容器。通过编译技术，依据构件的元数据信息、方面构件的元数据信息和构件与 XML 的编织信息，我们可以生成容器代码，在生成容器的源代码中，通过构件包容，实现方面对构件功能的横切。如对于 2.3 中的切割关系，将生成_Hello_容器，在_Hello_容器中包容 Hello 构件与 Greeting 构件，并且_Hello_构件的 IHello 接口的 SayHello 方法的实现代码中，将首先调用内部包容的 Hello 构件的 IHello 接口的 SayHello 方法，然后调用内部包容的 Greeting 构件的 IGreeting 接口的 AfterGreeting 方法，从而实现 XML 中描述的横切关系。通过对容器源码的编译，实现容器的二进制代码。另一种方法是在构件运行时，动态获得构件的元数据信息和方面构件的元数据信息，依据 XML 编织描述，动态的生成容器。第二种方法需要构件的二进制代码中封装构件元数据并通过构件的聚合技术实现。

2.4.3 容器的注册

为了实现运行时，对构件的调用转为对相应容器的调用，我们需要在构件系统中提供一种注册机制，记录构件与容器的对应关系（使用运行前容器生成技术），或者记录构件与描述构件与方面编织关系的 XML 文件的对应关系（使用运行时容器动态生成技术）。在我们的方法中，我们通过注册的方式，将这种对应关系记录在注册文件中。

3. CAR 中的面向方面编程的实现

我们在科泰的 CAR 平台上，初步实现了我们的方法。

3.1 CAR 中构件与方面构件的实现

CAR 作为一个面向构件的编程方法，它表现为一组编程规范，包括构件、类、对象、接口等定义与访问构件对象的规定。CAR 提供了一种构件描述语言 (CDL, Component Description Language)，CDL 与 COM 的 IDL(接口描述语言)相似，但是更为简洁，容易理解。在 CAR 中，使用 CDL 定义构件与方面构件，通过 CAR 的构件开发支持，能很方便的实现构件与方面构件。

3.2 CAR 上构件与方面的运行时动态编织

CAR 在生成构件时，将构件的元数据信息打包到构件的二进制代码中。因此，当系统接到客户端初始化某个构件的请求后，可以从构件的二进制代码中，根据元数据协议，从元数据信息中还原出构件的对象、接口函数及其参数等信息。根据这些信息，生成容器，可以保证容器的接口和服务端构件接口的表现完全一致.

我们在 CAR 系统上增加一个注册文件 aspects.reg，描述构件与方面的对应关系，aspects.reg 中记录构件信息，构件与方面的编织信息(XML 编织描述)。我们修改 CAR 的运行库，使得当一个构件被创建时，将首先检测 aspects.reg，如果该构件需要被方面横切，将根据其中的编织描述与该构件中的元数据信息，动态生成容器。容器内通过构件的聚会,实现构件与方面的结合，由容器负责转发客户所有的请求,并负责返回结果给构件的调用者，完成方面对构件的横切。

4 总结

本文提出了一种在构件平台上支持面向方面编程的方法，并在 CAR 上做出了初步实现。相比现有的方法，方面的定义与实现简单，不需要引入新的编程语言；支持现有构件的使用，不需要对现有构件做出修改；构件与方面的组合关系通过 XML 语言描述，通过容器实现运行时组合，具有比较高的效率。

需要说明的一点，作为研究课题，我们实现了系统的初步模型。如容器的静态生成，CAR 运行库的扩充。但对于实现复杂的切入关系的支持处理以及容器的动态生成需要在后续的课题中继续完成。

5 致谢

在课题的工作中，感谢陈榕老师，殷人昆老师的指导。感谢上海科泰世纪公司杜永文博士，刘亚东工程师的帮助。同时感谢我的同学范典，郭强，杨振宇对论文工作的帮助。

参考文献

[1] Davy Suvée, Wim Vanderperren, Viviane Jonckers, JAsCo: an aspect-oriented approach tailored for component based software development, Proceedings of

the 2nd international conference on Aspect-oriented software development, March 2003
[2] Kiczales, G., Lamping, J., Lopes, C.V., Maeda, C.,Mendhekar, A. and Murphy, A. Aspect-Oriented Programming. Proceedings of the 19th International Conference on Software Engineering (ICSE), Boston, USA. ACM Press. May 1997 Elisa L.A.
[3] Davy Suv´ee 1, Wim Vanderperren 2, Dennis Wagelaar and Viviane Jonckers: Towards a symbiosis between Aspect-Oriented and Component-Based Software Development
[4] Baniassad ,Gail C. Murphy, Christa Schwanninger and Michael Kircher: Managing Crosscutting Concerns During Software Evolution Tasks: An Inquisitive Study . AOSD 2002, Enschede, The Netherlands.
[5] JBOSS Group, “JBoss/AOP website,” Http://www.jboss.org.
[6]Davy Suv´ee 1, Wim Vanderperren 2, Dennis Wagelaar and Viviane Jonckers: There are no aspects. Electronic Notes in Theoretical Computer Science. 2004.
[7] M. Pinto, L.Fuentes, M. E. Fayad, J.M. Troya, Separation of Coordination in a Dynamic Aspect-Oriented Framework, Next Publication in the 1st International Conference on Aspect-Oriented Software Development, April 2002, The Netherlands
[8] Koretide Website, http://www.koretide.com.cn
[9] Koretide. CAR’s manual. 2004.
[10] AspectJ Website. Http://www.aspectj.org
[11] Frédéric Duclos, Jacky Estublier, Philippe Morat, Describing and Using Non Functional Aspects in Component Based Applications, Proceedings of the 1st international conference on Aspect-oriented software development, April 2002
[12] COM 本质论，（美）Box, D, 潘爱民译，中国电力出版社，2001
[13] COM 原理与应用，潘爱民，清华大学出版社，1999

作者简介：
李洪涛（1979－），男，硕士研究生，清华大学深圳研究生院软件工程中心，主要研究方向：
构件技术，基于构件的嵌入式浏览器
郑任持（1979－），男，硕士研究生，清华大学深圳研究生院软件工程中心，主要研究方向：
构件技术，基于构件的设备驱动
陈瑢（1957－），男，清华大学信息技术研究院操作系统与中间件中心副主任，科泰世纪科技有限公司首席科学家，主要研究方向：网络操作系统，构件技术
殷人昆（1945-），清华大学计算机系软件研究所副教授，软件工程实验室主任，主要研究方向：软件工程、数据结构、信息管理系统