一种面向方面的软件组件监控模型①

吴开贵，朱来雪

(重庆大学 计算机学院，重庆 400044)

摘 要：针对传统软件监控方法中模块化不好、缺乏灵活性的缺点，文中的模型中，将组件技术中的概念和方法应用于面向方面的技术中，提出了一个模块化、灵活性好的软件监控模型。在模型中将监控功能封装为一个面向方面的组件，有效解决了传统监控方法中因在应用代码中插入监控功能代码所产生的代码混乱与分散问题，也避免了因在组件和方面两个维度上考虑监控问题所产生的代码混乱与分散问题。

关键词：组件技术；AOP 技术；软件监控；织入；方面

Software Monitoring Model About Component and AOP Technology

WU Kai-Gui, ZHU Lai-Xue

(College of Computer Science, Chongqing University, Chongqing 400044, China)

Abstract: In this paper, a model for software monitoring is proposed by taking advantages of component technology and AOP (Aspect-Oriented Programming) technology. This model for monitoring is modular and flexible with notions of component at the level of AOP. The function of monitoring is encapsulated into an aspect component in the model. In this way, the code tangling and the code scattering issues caused by monitoring code inserted into function code are resolved.

Keywords: component technology; AOP technology; software monitoring; weaving; aspect

1 引言

随着软件技术的不断发展和人们生活对应用软件需求的不断增长，软件系统越来越趋于大型化和复杂化，在一个系统中，往往集成了若干个第三方组件。与此同时，基于组件的软件开发技术越来越成熟，采用该开发技术，使得代码的重用率得到很大提高，简化了软件开发过程，缩短了开发周期。因此，目前有很多大型软件系统采用了基于组件的软件开发方法，如企业信息门户系统(EIP)等等。

在一个基于组件的软件开发方法开发的软件系统中，系统是由若干个功能组件组合而成的，如用于解析 XML 文件的 SAX(Simple API for XML)组件，以及用于生成、读取 PDF 文件的 PDFBox 组件等等。组成应用软件系统的每个组件的性能对于该系统的整体性能是至关重要的，但是，要对所有的组件做测试又是不太现实的，因为商业组件没有提供测试用例，这样的组件对用户来说完全是一个黑盒。另外，每个组件都能正常工作，并不能保证该应用系统的正常工作。大量研究实验表明[1-5]，对应用软件进行监控，可以有效检测每个组件以及整个软件系统中的错误和异常，保证该系统的正常运行。

监控属于非核心业务，传统的软件监控方法是将监控代码插入到核心业务代码之中，这样容易打乱核心代码的业务逻辑，引起代码纠缠和分散，不利于以后的软件维护工作。监控是一个经典的横切关注点问题，因此，应用 AOP(Aspect-Oriented Programming)技术可以实现监控代码与监控对象代码的分离，这样实现的监控模型灵活性和扩展性都很好，而且更加模块化[5]。应用 AOP 技术来实现对由组件组合成的系统的监控，就需要考虑组件技术与 AOP 技术结合的问题。

① 基金项目:国家自然科学基金(90818028)
收稿时间:2010-07-26;收到修改稿时间:2010-08-25

将 AOSD（面向方面的软件开发方法）应用于 CBSD（基于组件的软件开发方法）来指导软件开发过程，已经有人提出[6]。但是这种单向结合的方法并不利于实现软件的监控，因为目前的 AOSD 方法都是非对称的，也就是说，从结构上来看，方面和组件是不同的实体，它们是按不同的规则组合起来的。这样就需要从两个维度来考虑问题，难以实现系统的维护和监控。

受到 Pessemier 等人工作的启发，本文吸取组件技术和 AOP 技术的优点，提出了一个基于组件技术与 AOP 技术的软件监控模型。该模型中，将组件技术中的概念方法应用于 AOP 技术中，设计了一个模块化、灵活的软件监控模型。在模型中，我们定义了监视器组件、监视域、监视器织入和功能组件，在后面的模块中会给出这些概念的详细解释。

目前对应用软件的监控研究主要集中在对服务器软件和数据库等重要的应用上，这类应用的运行状况至关重要，因此对其进行的监控研究也就比较深入，而对那些普通的应用软件，相应的研究也就少的多。Barnett 和 Schulte 在 2003 年根据可执行规范实现了一个运行时监视器模型，该模型中使用模型编程(Model Program)描述组件间的不同属性间的交互，但是，该模型没有考虑到并发性，因此不能处理多线程组件的情况[4]。伊利诺斯大学的 Feng Chen 和 Grigore Rosu 提出了一个 MOP（Monitoring-Oriented Programming）模型，通过在目标系统中插入注释，来获得软件的运行时信息。但是这种基于底层代码实现的软件监控模型，它的监控层次低，灵活性差[1,2,9]。

2 监控模型

2.1 相关技术介绍

2.1.1 AOP 技术[5,6]

AOP 技术拥有一些重要的术语和概念，彻底理解这些术语和概念是掌握 AOP 技术的关键。以下对 AOP 中最重要的术语进行解释。

切面(Aspect)：所谓的切面，就是横切关注点的模块化，是在传统程序设计方法学中难于清晰地封装并模块化实现的设计决策，封装实现为独立的模块。在我们的模型中，切面表现为组件。组件具有可重用的特性，因此，在我们的模型中的切面也具有可重用性。

建议(Advice)：所谓建议，就是方面的代码实现，即在满足条件的连接点处运行的一段代码，也就是执行逻辑。执行逻辑可以在连接点本身、之前或之后被运行，如 AspectJ 根据执行逻辑运行的位置，提供了三种建议类型，before、after 和 around。

连接点(Join point)：所谓连接点，就是程序流程中某个可以被执行的点或位置。方面在连接点处与主程序交互。连接点可以是方法的调用、异常的触发和变量的赋值。

切入点(Pointcut)：所谓切入点，就是连接点的集合，它定义了程序流程中的若干连接点。

织入(Weaving)：织入是一个动态过程，就是将方面代码利用方面编织器织入到核心代码中，以便在适当时候触发建议代码的执行。

2.1.2 组件技术[7,8]

从组件的组成来看，组件是指具有契约化定义的接口以及明确的上下文依赖关系的可组合程序单元。从组件的用途来看，组件是指可以被高度重用的软件单元，它封装了一定的数据、属性和方法，可以被独立地部署和提交给第三方进行组合。组件遵循二进制外部接口标准，内部实现细节对用户透明，具有即插即用的特性。一个基于组件的软件系统，结构往往如图 1 所示。

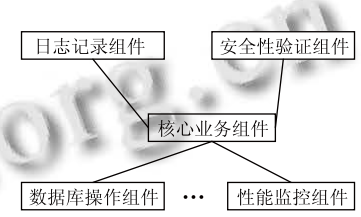

图 1　基于组件的软件系统结构

组件具有以下特性：

(1) 组件具有明确的上下文依赖关系,在一个组件中不仅提供了特定的服务，并且在实现其服务的过程中也有可能调用其他组件提供的服务，组件至少应当包括对外提供的服务。

(2) 组件提供的接口是服务提供者与服务使用者之间关于服务及其用法的契约，这些接口使得组件具有良好的可扩展性。

(3) 组件的实现细节对用户透明,组件对外表现为一个“黑盒” 结构。这样，使得组件可以被不同的开

发商独立地生产、获取和配置，应用不同的组件来搭建应用程序。

在一个基于组件的软件系统中，各组件间不是孤立的，而是相互交互、相互影响的。组件之间的交互通过组件提供的接口来实现，接口提供了组件交互所需的全部信息。接口是对组件提供和请求服务的抽象描述,是组件服务及其用法的契约,独立于任何特定的实现。

2.2 监控模型的结构

2.2.1 模型中的重要概念

本文吸取组件技术与面向方面技术的共同优点，设计了一个基于组件技术与面向方面技术的软件监控模型。该模型中有三个重要概念：监视器组件、监视器织入和监视域。

监视器组件是组件技术与 AOP 技术的优点相结合产生的。监控是一个横切关注点问题，我们将该横切关注点的建议代码模块化，封装在一个或多个组件中就形成了监视器组件。各个建议代码通过组件的接口向外提供服务。监视器组件其实就是一个面向方面的组件[6]。图 2 代表的就是一个监视器组件模型，该模型提供两种接口，一种是向外提供服务的输入接口，另一种就是调用监控目标，接收返回信息的输出接口。通过 AOP 技术的织入机制，将监视器组件织入到要监控的目标系统中，实现监控功能。一个组件要想成为监视器组件，它必须至少提供一个建议调用接口。

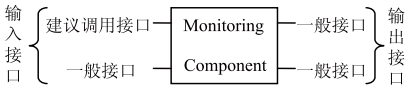

图 2 监视器组件

监视器织入就是将定义的监视器组件应用到目标组件上，搜集组件运行时信息，以实现对目标组件的监控，保证软件系统的正常安全运行。监视器织入是依靠 AOP 技术的织入机制来实现的。监控目标组件提供一组连接点，监视器组件就可以在这些连接点处被织入，达到监控目标组件的目的。

监视域就是监视器组件所监控的组件范围。通过监视域，我们可以清楚的看出哪些组件处于被监控状态。

监视器的织入过程通过切点语言(Pointcut language)[6]实现。切点语言由关键字和三个正则表达式组成。关键字可以是 SERVER，也可以是 CLIENT，分别代表输入和输出接口，关键字为空时，代表输入输出接口都可以。三个正则表达式之间用分号隔开，分别代表组件、接口和方法，即 AOP 技术中的切点。下面是切点语言的语法结构：

pcd:=jp_type component; interface; method

jp_type: CLIENT|SERVER|BOTH

component:关于组件名字的正则表达式

interface: 关于接口名字的正则表达式

method: 关于方法名字的正则表达式

下面是几个切点表达式的例子：

*; *; add*: void 这个切点表达式表示所有组件中返回值为空的以 add 开头的输入输出方法。

CLIENT B; *; add* 这个切点表达式表示组件 B 的接口中名为 add 的输出方法。

SERVER B; DAO; * 这个切点表达式表示组件 B 中名为 DAO 的所有输入方法。

切点表达式用于将监视器组件与要监控的目标组件绑定起来，实现对目标组件的监控。当一个组件中接口的名字与切点表达式中的正则表达式匹配时，该组件就处于监控之中。

2.2.2 监控模型结构

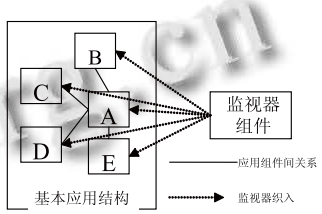

图 3 监控模型框架

图 3 是监控模型的基本结构，它主要由两部分组成：基本应用系统和监视器组件。图 3 的左边是基本应用系统的结构，由 A、B、C、D 和 E 五个组件组成；图的右边为监视器组件，它实际上是一个面向方面的组件，实现对应用系统的监控功能。图 3 中的实线表示基本应用系统中组件间的关系，虚线表示监视器与基本应用中各个组件间的关系，即监视器的织入。

当要对基本应用进行监控时，我们用切点表达式来描述监视器与应用系统中各个组件间的关系，选择对应用系统中的哪些组件以及组件中的哪些操作进行

监控，然后由 AOP 技术的织入机制，将监视器与基本应用联系起来，实现对应用系统的监控。

2.2.3 监视器组件的内部结构

监视器组件是该监控模型的核心，它负责搜集软件系统的运行时信息，分析软件系统的运行状况，并将分析结果呈现给用户。这样用户就可以时刻掌握应用系统的运行状况，保证系统的正常运行。即使应用系统出现了错误，用户也能快速找到问题所在，使应用系统在最短的时间内恢复正常运行。

图 4 是监视器组件的内部结构，它由一个绑定配置文件(binding.xml)、绑定器、监视器和分析器组成。绑定配置文件是一个 XML 文件格式的文件，我们可以在这个文件中配置应用系统中哪些组件需要监控，组件中哪些方法需要监控，还可以指定监控组件的哪些方面，如一次请求的执行时间等。这些指定是通过切点表达式来完成的。绑定器解析刚才定义的绑定配置文件，将监视器织入到应用系统中，使得应用系统在特定的位置触发监视器的执行。监视器通过在特定的时刻执行，搜集软件系统的运行时信息，传给分析器。分析器分析监视器传过来的信息，将分析结果呈现给用户。如在 J2EE 应用中，可以通过 web 页面呈现给用户，也可以通过 Java 管理扩展(JMX)呈现给用户。

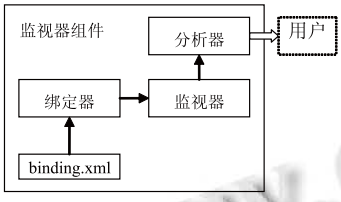

图 4　监视器组件内部结构

2.3 监控模型分析

将组件技术与 AOP 技术结合起来研究软件监控技术，具有以下优点：

(1) 逻辑结构清晰，避免了代码纠缠不清的问题。在软件监控中，代码纠缠问题由两个原因产生的：一个是在传统的软件监控方法中，另一个是在基于组件的软件开发方法中。在传统的软件监控方法中，我们需要在代码中设置监控代理，这样监控代码与业务代码混在一起，造成代码混乱。而在基于组件的软件开发方法中，一个组件中有多个关注点，一个关注点又会在多个组件中，这样同样会产生混乱，并且代码重用率不高。本文在基于组件的软件开发方法中，吸取 AOP 技术的优点，将监控功能抽象为一个面向方面的组件，通过 AOP 的织入机制，将监控功能织入到应用代码中，就不会出现代码纠缠不清的问题，并且代码的重用率也得到了提高。

(2) 抽象层次高。在本文的模型中，通过配置文件，描述组件提供的接口与监控组件之间的关系，搜集软件运行时信息，而不必到代码中设置监控代理。因此我们只需要了清楚组件提供的接口，而不必知道组件的内部关系，抽象层次由代码层提高到了组件层。

(3) 实施简单。监控功能以组件的方式实现，并提供一个配置文件，来描述目标系统与监控组件之间的关系。当要对目标系统进行监控时，只须修改配置文件，而不必知道目标系统与监控组件的内部结构，因此实施简单。

3 结论

本文吸取组件技术与 AOP 技术二者的优点，提出了一个基于组件技术与 AOP 技术的软件监控模型。该模型具有模块化、易扩展的优点。在模型中将监控功能封装为一个面向方面的组件，通过切点表达式描述监视器组件与应用组件之间的关系，然后将监视器织入到目标系统中，有效解决了传统监控方法中因在应用代码中插入监控代码所产生的代码混乱与分散问题。组件和方面是两个维度上的实体，把监控功能封装为面向方面的组件，就避免了因在两个维度上考虑问题所产生的代码混乱与分散问题。目前，该模型的框架、内部结构和运行原理已经得到了分析论证，下一部主要工作是根据该监控模型，实现一个监控系统，将该系统应用于监控实际的软件系统，保证软件系统的正常运行。

参考文献

1 Chen F, Rosu G.. Towards monitoring-oriented programming: a paradigm combining specification and implementation.

Electronic Notes in Theoretical Computer Science, 2003,89(2):108−127.

2 Chen F, Rosu G. MOP: An efficient and generic runtime verification framework. Proc. of the 22nd annual ACM SIGPLAN Conference on Object-oriented Programming Systems and Applications. Conference on Object Oriented Programming Systems Languages and Applications, Montreal Quebec Canada. New York: ACM, 2007.569−588.

3 张玮.组件技术在门户系统中的应用研究[硕士学位论文].南昌:南昌大学,2007.

4 Barnett M, Schulte W. Runtime verification of .NET contracts. The Journal of Systems and Software, 2003, 65: 199−208.

5 徐理.基于 AOP 的应用软件监控技术研究[硕士学位论文].长沙:国防科学技术大学,2008.

6 Pessemier N, Seinturier L, Duchien L. A component-based and aspect-oriented model for software evolution. Int. J. Computer Applications in Technology, 2008,31(1/2):94−105.

7 Zulkernine M, Seviora R. Towards automatic monitoring of component-based software systems. The Journal of Systems and Software, 2005,74:15−24.

8 秦宇,冯登国.基于组件属性的远程证明.软件学报,2009,20(6):1625−1641.

9 Chen F, Jin DY. Monitoring oriented programming-a project overview. International Conference on Intelligent Computing and Information Systems. Ain Shams University, Cairo, Egypt, 2009:72−77.