一种动态监测模型的研究

黄 隽，金可音

（湖南工业大学 计算机与通信学院，湖南 株洲 412008）

摘 要：在开放动态的网络环境中，分布式软件呈现出规模庞大、松散聚合、行为复杂等特点，需要对其进行动态监测。提出了一种由策略驱动的监测模型，该模型由适应性策略选择AOP切面，监测模型的监测器由这些切面组成。利用动态织入机制，在目标系统运行过程中，随着监测值的变化，适应性策略动态地增加或删除监测器。因此，监测模型能自适应程序和当前系统的变化，动态调整监测强度，以使监测负载保持均衡，从而取得最佳监测效果，达到动态监测的目的。模拟实验结果表明，所设计的监测模型是合理的。

关键词：动态AOP；策略；动态监测

中图分类号：TP302 **文献标志码**：A **文章编号**：1673-9833(2011)02-0055-04

Research of A Dynamic Monitoring Model

Huang Jun，Jin Keyin

（College of Computer & Communication, Hunan University of Technology, Zhuzhou Hunan 412008, China）

Abstract： In an open and dynamic network environment, distributed software shows the characteristics of large-scale, loose aggregation and complex behavior, which needs dynamically monitor. Proposes a strategy-driven monitoring model in which AOP (Aspect Oriented Programming) aspects are selected by adaptive strategy. With dynamic weaving mechanisms, the monitor composed by the aspects can be added and deleted according to the change of monitor value in the running process of target system. Therefore the model is able to adapt procedures and current system conditions changing and dynamically adjust the monitoring intensity to keep a balance of monitoring load, so as to get best monitoring results and achieve dynamic monitoring purposes. Simulation results show that the design of the monitoring model is reasonable.

Keywords：dynamic AOP; strategy; dynamic monitoring

1 相关研究现状

软件可信是指软件系统总是按照其设定目标所期望的方式运行[1]。软件本质上是代替人执行一定的行为，软件的可信性主要表现在其行为可信上，即要求软件系统的运行行为及其结果总是符合人们的期望[2]。软件的可信性不是凭空而来的，需要对其进行监测，通过监测可得到软件系统的运行实况和相关数据，并以此为基础，对软件进行容错处理、故障诊断和性能优化等处理，以最终提高软件的可信性。因此，如何对软件进行有效监测，是一个具有理论意义和应用价值的研究课题。

国内外已对系统软件的运行监测开展了一系列的研究，出现了一些较为成熟的监测技术和方法。

收稿日期：2010-12-08

作者简介：黄 隽（1981-），男，湖南长沙人，湖南工业大学硕士生，主要研究方向为计算机软件及理论，
E-mail：publicthing@126.com

如文献[3-4]提出了一种分布式组件软件的运行监测机制，研究者将监测代码封装成一个个的组件，然后利用组件来监测软件的性能、状态和交互事件，并收集软件运行时的信息。文献[5]使用构件包装器来自动捕获程序运行行为，这种方法适用于第三方构件，缺点是监测开发人员必须人工编写大量代码，因此对开发人员的编程能力有一定的要求。文献[6]使用程序插桩技术在程序中插入监测代码，插桩后的程序在运行过程中，能实时地将软件运行信息传递给监测器，以达到监测系统的目的。文献[7-8]根据给定的分布式软件行为规约自动生成观察代码，动态监测系统运行行为。这种技术直接将监测代码插入源代码中，其优点是监测负载较低，但是这种方法需要获得目标软件的源代码，不适合于主要由无源代码的黑盒实体构成的分布式软件。在基于面向切面编程（aspect oriented programming，简称AOP）的监测技术研究方面，文献[9]将AOP技术应用到软件的运行轨迹监测中，为系统故障诊断提供量化依据；文献[10]对分布式多线程系统进行了研究，开发出了一套监测框架以及相应的监测工具，该监测框架采用全局跟踪技术，通过编译器自动生成探针，并将探针植入目标系统中。

由此可知，经过长期的科研与实践，对分布式软件监测领域的研究已取得了一定的进展，形成了一些为多方所一致认同的监测技术，开发出了很多实用的监测工具。但是通过分析，笔者发现上述研究中还存在如下不足：1) 大多数研究没有提供一套良好的监测机制，监测逻辑与业务逻辑紧密耦合，大量的监测代码散布于业务逻辑代码中，给程序维护带来了困难；2）大多数监测系统都不支持用户动态配置和自主调整监测需求，监测代码被编写成“硬代码”，监测规模不能随着目标系统规模的扩展而扩展，监测代码自身的修改也不方便。

针对分布式软件监测中存在的这些不足，笔者拟设计一套可伸缩性较强、可扩展性较好，可靠性较高和监测负载较低的监测框架。

2 监测模型设计

2.1 构件交互行为监测的分类

随着中间件技术的成熟，基于中间件的分布式构件软件得到越来越广泛的应用，对于这类分布式软件的交互行为监测显得尤为重要。构件交互行为监测是指对构件在交互活动期间所进行的操作进行监视记录的过程。根据对构件交互行为可信性关注点的不同，具体确定对哪些感兴趣的行为信息进行监测，以控制合适的监测强度。例如，在电子商务应用系统中比较关注交互行为的时效性、安全性等，那么除了监测交互行为的基本信息外，还要对与交互行为的时效性、安全性等密切相关的数据进行监测。依据获取的监测数据不同，构件交互行为监测可分为如下几类：

1）交互行为基本信息监测。监测的内容有事件名、事件类型、线程的状态信息、CPU使用率、内存使用率等。

2）交互行为时效性监测。监测的内容有交互事件开始执行的时间、交互事件结束执行的时间等。

3）交互行为安全性监测。监测的内容为构件遭受恶意攻击的次数。

关于构件交互行为的时效性、安全性等可信性质的描述，由于覆盖面太广、标准太多，很难提供一致公认的标准，故针对不同的应用场景有所不同。

2.2 监测模型的总体结构

分布式软件由多个软件实体松散聚合而成，软件实体以开放、自主的方式存在于网络的节点中，鉴于分布式软件的特点，本文设计了一种层次化的监测框架，如图1所示。

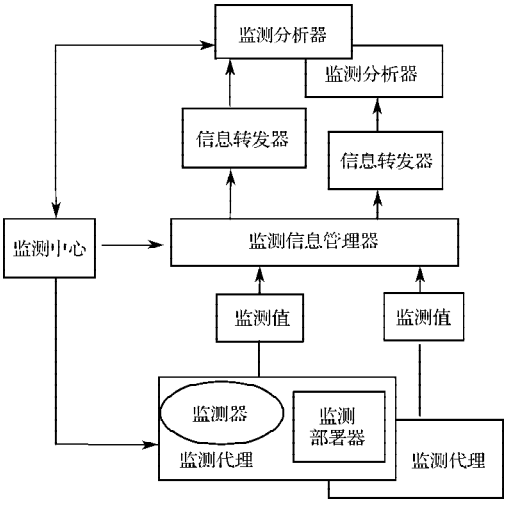

图1 监测模型总体结构

Fig. 1 Architecture of monitoring model

监测框架设计的基本思想如下：

1）监测器负责监测程序，获取监测信息。

2）监测信息管理器负责收集所有监测器捕获到的监测信息，并经过预处理后将结果信息传输至转发器。

3）监测分析器接收转发器传来的监测信息，并生成监测定义描述文件，再传送给监测部署器，由其对监测器进行部署。

4）监测中心全面管理模型中的各个监测部件，

提供可视化界面，让用户动态配置监测需求和查看监测结果。并生成适应性策略文件，传输给监测分析器。配合分析器，传递监测定义描述文件给监测部署器。控制监测信息管理器，创建合适的信息转发器。

2.3　监测模型的组成部件

监测器　监测器是监测代理的核心部件，也是整个监测模型的关键部件。借助动态 AOP 技术，任意一个实现了预定义接口的切面，都可成为获取目标系统监测信息的监测器。使用动态 AOP 技术的监测器主要有如下特点：1）遵循关注点分离原则，监测逻辑与业务逻辑相互分离，功能模块和监测模块是“正交”的，监测代码横切入功能代码中，即监测器能以更灵活、松散、透明的方式融入目标系统。2）利用动态织入机制，能在不改变被监测系统源代码的前提下，在目标系统的运行过程中将监测器动态织入待监测构件中。该处理不但解决了对黑盒软件无法获悉源代码的困难，而且监测器的植入与撤除是一种“热插拔”的方式，对被监测实体的功能不会造成任何干扰。既便于系统进行动态配置和调整监测需求，也便于选择不同类型的监测器动态地织入或撤出目标系统，以达到控制监测强度的目的。

监测部署器　监测分析器生成监测定义描述文件后，传输给监测部署器。监测部署器对文件进行解析后，先查找监测代理里有没有合适的监测器切面代码，如果没有，可远程调用切面代码。查找到合适代码后，再生成监测器配置文件，将监测器动态织入待监测构件上。依据设定的拦截点 before advice/around advice/after advice 等，在构件方法调用之前 / 中 / 后等进行拦截，动态地执行监测代码。

监测信息管理器　监测器获取的监测值全部发给监测信息管理器。管理器将监测信息经过预处理后，传输给信息转发器。监测信息管理器还具有接收监测中心的指令，添加或移除信息转发器的功能。

信息转发器　信息转发器起到了一种监测信息转换的作用。它可以按监测分析器需要转换监测数据的格式。当分析器与信息管理器之间的传输网络负载比较高时，转发器还能暂时保存监测信息，这样既保证了监测数据不会丢失和监测分析器能随时获取必要的数据，还能尽量缓解性能瓶颈。

监测分析器　监测分析器接收监测中心传来的适应性策略，并通过对所接受的策略进行分析后，产生监测定义描述文件。然后与监测中心合作，控制监测部署器对监测器的配置。监测器开始监测后，将监测信息发给监测分析器，分析器对照适应性策略，看是否对当前的监测状态进行改变。当监测状态要改变时，监测分析器产生新的监测定义描述文件，并将新的监测器部署到要监测的软件实体上。监测分析器与监测代理和被监测程序之间的关系如图 2 所示。

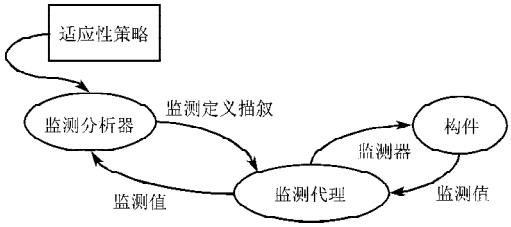

图 2　监测循环控制图

Fig. 2　Monitoring cycle control

监测中心　监测中心从总体上负责监测代理、监测信息管理器和监测分析器的管理。它的具体功能有：提供可视化界面，让用户动态查看监测结果；利用反射技术提取出目标系统中构件的内部结构信息，暴露出待监测对象；接收用户动态配置的监测需求信息；生成适应性策略文件，并传输给监测分析器；与分析器合作，控制监测部署器，动态配置监测器，使得监测负载保持均衡；通知监测信息管理器准备接收监测代理传来的监测值，并要求管理器创建好合适的信息转发器。

2.4　动态监测模型的监测过程

监测模型中，各部件相互协作，共同完成监测任务。动态监测的完整过程可描述如下：

1）监测中心借助反射技术，暴露出目标系统中所有待监测对象。在明确需要监测的交互行为类型后，用户动态配置监测需求。将根据监测需求生成的适应性策略文件传输给监测分析器。监测分析器参照适应性策略生成监测定义描述文件，经由监测中心转发给监测代理。同时，监测中心通知监测信息管理器准备接收监测信息，并且创建合适的信息转发器。

2）监测代理中的监测部署器对监测定义描述文件进行解析后，根据定义匹配合适的监测器，生成监测器配置文件，动态织入到待监测对象上。监测器截获监测信息后，发送至监测信息管理器。

3）监测信息管理器接收监测信息后，经过预处理，将监测值分派给合适的信息转发器，由信息转发器将监测信息传送给对应的监测分析器。

4）监测分析器将接收到的监测值与适应性策略中的控制规则进行对照。当监测值没有超过控制规则中的阀值时，维持当前的监测状态，分析器等待下一次的监测值送达；当监测值超过阀值时，表明需要改变当前的监测状态，监测分析器产生新的监

测定义描述文件，通过监测中心转发给监测代理，动态地增加或删除监测器，开始新的一轮监测。

3 实验与结果分析

为了验证所设计模型的合理性，拟对系统的动态监测过程进行仿真模拟实验。设计的待监测程序用来产生TestObject对象，每个生成的TestObject对象用ID号来区分，每当getTestObject()方法调用时，就产生一个TestObject对象。getTestObject()方法调用的频率逐步提高，当方法执行时间超过规则设定阀值（时间信息由MeasureMethodTime切面提供），cache切面被动态织入监测目标，用来加快TestObject对象生成速度。所得测试结果如图3所示，从图3可看出，织入cache切面后，程序执行速度有了较大的提高。因此，该动态监测的实施是成功的。

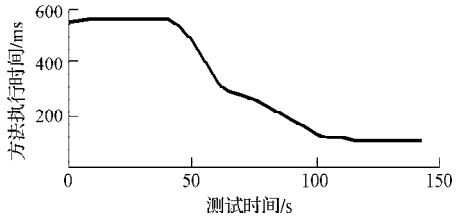

图3 测试效果

Fig. 3 Testing result

4 结语

针对分布式软件监测领域存在的主要不足之处，提出了一种动态监测模型，该模型包括监测代理、监测中心和监测信息管理3个模块，并设计了各监测部件的功能和监测流程。模拟实验结果表明了该动态监测模型的可操作性。在今后的工作中，将进一步完善实验，并改进动态监测模型在分布式软件监测中的应用。

参考文献：

[1] 陈火旺，王 戟，董 威. 高可信软件工程技术[J]. 电子学报，2003，31(12)：1933-1938.
Chen Huowang，Wang Ji，Dong Wei. High Confidence Software Engineering Technologies[J]. Acta Electronica Sinica，2003，31(12)：1933-1938.

[2] 王怀民，唐扬斌，尹 刚，等. 互联网软件的可信机理[J]. 中国科学：E辑：信息科学，2006，36(10)：1156-1169.
Wang Huaimin，Tang Yangbin，Yin Gang，et al. Trustworthiness Principle of Network Software[J]. Science in China：Series E：Information Sciences，2006，36(10)：1156-1169.

[3] Jerry G，Zhu Y，Shim S. Monitoring Software Components and Component-Based Software[C]// Proceedings of the 24th IEEE Annual International Computer Software and Applications Conference. Taiwan：IEEE Computer Society, 2000：403-412.

[4] Krumm H. Trust-Adapted Enforcement of Security Policies in Distributed Component Structured Applications[C]// Proceedings of the 6th IEEE Symposium on Computers and Communications. Tunisia：IEEE Computer Society，2001：2-8.

[5] Mariani L，Pezzè M. A Technique for Verifying Component-Based Software[J/OL]. [2010-10-20]. http：//www.docin. com/p.87315999.html.

[6] Kim M. Information Extraction for Run-Time Formal Analysis[D]. Philadelphia：University of Pennsylvania，2001.

[7] Logean X，Dietrich F，Hubaux J，et al. Run-Time Monitoring of Distributed Applications[C]//Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware' 2000). USA：Springer，2000：459-473.

[8] Chen F，Rosu G. MOP: An Efficient and Generic Runtime Verification Framework[C]//Proceedings of the 2007 ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA 2007). Canada：ACM，2007：569-588.

[9] 张瞩熹，王怀民. 基于AOP的软件运行轨迹捕获技术研究与实现[J]. 计算机应用，2008，28(5)：1322-1324.
Zhang Zhuxi，Wang Huaimin. Research and Implementation of Trace Capture Technique Based on Aspect-Oriented Programming[J]. Computer Applications，2008，28(5)：1322-1324.

[10] Li J. Monitoring and Characterization of Component-Based Systems with Global Causality Capture[C]//Proceedings of the 23rd International Conference on Distributed Computing Systems. [S. l.]：IEEE Computer Society，2003：422-431.

（责任编辑：廖友媛）