

一种基于工作流自适应软件系统的监控模型

章小卫¹, 顾梦远², 李 斌¹

(1 扬州大学 信息工程学院, 江苏 扬州 225127; 2 南京大学 软件学院, 江苏 南京 210018)

摘 要: 本文提出了一种针对工作流的系统监控框架. 该框架在模型层面赋予了用户更灵活的表达方式, 在实例化的运行层面提供了动态的指标计算和具体的实施, 既能实时监控系统运行状况, 又能监控系统上下文环境, 从而使得软件系统能够更好的满足用户需求的变化以及运行环境的变更.

关键词: 监控; 自适应软件系统; 上下文; QoS

中图分类号: TP391

文献标识码: A

文章编号: 1000-7180(2012)11-0025-04

A Monitoring Model of Workflow Based Self-adaptive Software System

ZHANG Xiao-wei¹, GU Meng-yuan², LI Bin¹

(1 College of Information Engineering, Yangzhou University, Yangzhou 225127, China;

2 College of Software, Nanjing University, Nanjing 210018, China)

Abstract: We proposed an system monitoring model of self-adaptive system based on our previous work, which gives users more flexibility in expressing requirements on the model level, provides dynamic QoS value calculation and implementation at run time, and can both monitors operating status of system in real-time and monitor the context of the system, allowing software systems to better meet users' changing needs, and the operating environment changes.

Key words: monitor; self-adaptive system; context; QoS

1 引言

随着 Internet 向社会各角落的渗透式扩张, 分布式计算、网格计算、普适计算等新型应用模式不断涌现. 软件系统的复杂性和规模不断增加, 软件系统所处的环境具有动态性、异构性以及时刻变化等特征, 软件系统如何对外部环境、外部和内部需求的变化做出及时并且正确的反应, 使得软件系统能够始终维持在最优的状态, 是软件开发面临的重大挑战. 针对上述挑战, 自适应软件^[1-3]提出了许多新的解决机制, 并逐渐成为当前的研究热点.

对于自适应软件系统的设计有许多方法, 其中包括基于 Agent 技术的方法^[4]和基于工作流^[5-6]的方法等. 由于 Web 服务技术的迅速发展, 并不断应用于社会各领域中, 基于工作流的自适应软件系统

设计方法得到很多研究者的关注. 然而, 无论采用哪种方法, 软件系统要实现自适应调整, 前提是必须能够在运行过程中, 及时的获得系统内、外部环境信息. 因此, 实时而有效的监控手段是软件系统实现自适应调整的不可或缺的关键步骤.

目前, 基于工作流的自适应软件系统的监控模块大多数都是监控 Web 服务信息^[7], 而这方面的研究正处于起步阶段, 目前已有成果大都侧重于系统的自适应调整策略的研究, 并未提出有效的监控方法, 有的即便是给出了监控模型, 但未给出领域相关的 QoS 进行定义和 QoS 值的计算方法, 可扩展性差, 存在很大的局限性.

针对现有工作的不足, 本文在我们既有工作的基础上, 借鉴了现有的工作流的监控和 QoS 采集方法提出了一种针对工作流可扩展的服务监控框架,

在模型的层面赋予了用户更加灵活的表达能力,在实例化的运行层面提供了动态的指标计算、以及具体的实施,从而能够更好的满足用户需求的变化以及运行环境的变更.

2 自适应软件系统监控框架

监控框架包括 Web 服务 QoS 指标监控和上下文监控两个子模块. Web 服务 QoS 指标监控子模块负责 QoS 信息的采集、计算和存储,上下文监控子模块负责采集流程执行时的上下文信息,并把上下文信息发送给自适应模块.

图 1 描述了监控框架的结构,此架构分为三个层次:Web 服务工作流程层、监控层、预测和自适应层. Web 服务工作流程是监控源;监控层是主体;预测和自适应层是监控数据的汇聚点. Web 服务工作流程层中的流程一旦开始执行,将触发监控层的两个子模块开始工作,它们分别采集 Web 服务 QoS 指标信息和上下文信息,并在对这些信息进行分析后,把分析结果发送给预测和自适应层,预测和自适应层收到监控层发送来的数据后,对照系统的目标对系统作出自适应调整.

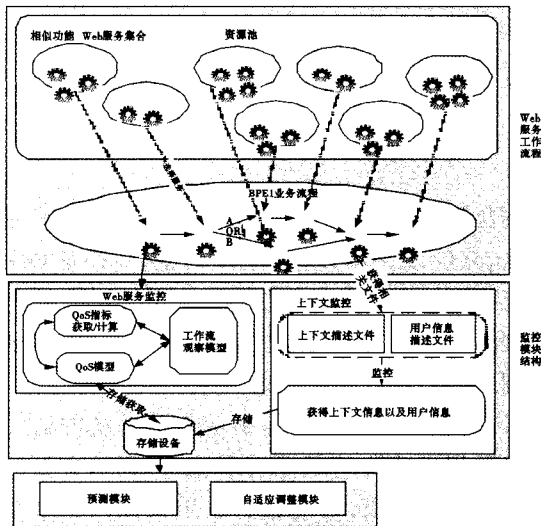


图 1 自适应软件系统监控框架图

3 Web 服务 QoS 指标监控

3.1 Web 服务 QoS 监控模块结构

图 2 描述了监控层的 Web 服务 QoS 指标监控子模块结构,其中包括 QoS 模型、工作流监控元模型、指标采集与计算模块以及 QoS 库,各个模块之间互相协作共同完成了对 Web 服务 QoS 指标的监

控。

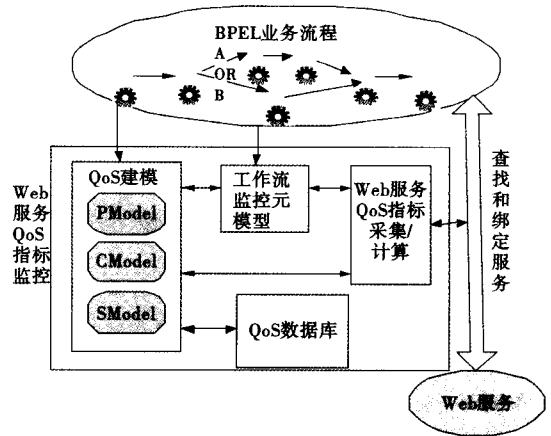


图 2 Web 服务监控框架图

(1) 工作流监控元模型是一个抽象的监控类,一旦某一个 BPEL 工作流开始执行,工作流监控元模型为它创建一个监控实例,并且触发 QoS 建模模块对该流程中的每一个 Web 服务的特定的 QoS 指标创建并实例化 QoS 模型. QoS 模型创建后,它还将触发 Web 服务 QoS 指标采集与计算模块,通过采集与计算获得 QoS 指标值.

(2) QoS 建模模块包括 PModel, CModel 和 SModel 三类 QoS 模型,它负责定义 QoS 指标,存储 QoS 指标值,创建并实例化 QoS 模型,并把对某一个服务的 QoS 指标的监控状态发送给其它模块.

(3) Web 服务 QoS 指标采集/计算模块的任务是采集并计算 QoS 指标,并把计算结果发送给 QoS 建模模块, QoS 建模模块把 QoS 指标值存储到 QoS 库中.

3.2 Web 服务 QoS 指标分类及模型定义

3.2.1 Web 服务 QoS 指标分类

根据获得 QoS 指标方法的不同,把它分以下三类:

(1) 供应级指标 (Provider-advised metrics)

① 由服务供应商给出的指标,并通过 QoS 指标采集/计算模块采集获得;

② 是由用户的需求决定,非系统必需的 QoS 指标;如:服务价格(prices).

(2) 消费级指标 (Consumer-rated metrics)

① 是由用户给出的 QoS 指标值;

② 可以通过存储设备获得已有的指标值;如:用户评价(evaluation).

(3) 系统级指标 (IT level metrics)

① 在系统执行过程中产生的 QoS 指标值;

② 在系统执行过程中,通过 QoS 指标采集/计算模块采集获得;如:执行时间(Runntime)。

3.2.2 QoS 模型定义

根据以上 QoS 的分类,本小节建立了相应 QoS 模型。图 3 描述了该模型图。我们把 QoS 模型(QoSModel)也分为三类:供应级指标 QoS 模型(PModel)、消费级指标 QoS 模型(CModel)、系统级指标 QoS 模型(SModel)。

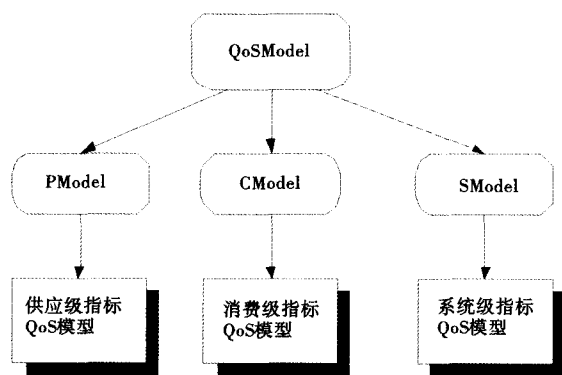


图 3 与 QoS 分类相应的 QoS 模型

(1) 供应级 QoS 指标模型,供应级 QoS 指标的属性包含了工作流 ID、指标名称、指标值等相关信息,提供了该类 QoS 指标的采集方式。

(2) 消费级 QoS 指标,消费级 QoS 指标模型主要包含了工作流 ID、服务 ID、指标的名称和指标值等相关信息,它的指标值是由用户直接给出的,例如用户评价。

(3) 系统级 QoS 指标,系统级 QoS 指标模型包含属性、计算规则和数据采集三个部分。属性同样包含了工作流 ID、指标名称、指标值等相关信息;计算规则包括输入数据、计算公式和指标值,其中输入数据是由采集计算子模块通过定义好的采集方式获得,并利用指标的计算公式计算出指标值;数据采集定义了需要采集的数据和采集方式,同时提供了该类 QoS 指标的采集过程。

3.3 工作流监控元模型

单一的监控模型无法满足所有种类业务流程的需要,所以必须为每一个 BPEL 业务流程建立相应监控模型。为此本文设计了一个工作流监控元模型。

工作流监控元模型其实就是一个抽象的类,通过实例化该类,可以为不同的 BPEL 业务流程创建不同的监控模型。它包括工作流、Web 服务和 QoS 指标三个模块,这三个模块分别对应 BPEL 业务流

程中的三个监控实体。在建立监控模型的过程中,工作流模块对应一条由系统规划好的 BPEL 业务流程;Web 服务模块对应于该流程中每一个服务节点的 Web 服务监控实体;QoS 指标模块对应于 Web 服务的 QoS 指标监控实体。

3.4 Web 服务 QoS 指标采集与计算

Web 服务 QoS 指标采集与计算子模块是 Web 服务 QoS 指标监控模块的核心。不同种类的 QoS 指标应采用不同的采集方式,因此采集模块中包括系统级 QoS 指标采集、供应级 QoS 指标采集和消费级 QoS 指标采集子模块。

(1) 系统级 QoS 指标采集子模块,包括 QoS 模型片段(SModel)、QoS 指标计算、采集数据存储以及数据采集四个部分。该子模块的结构如图 4 所示。

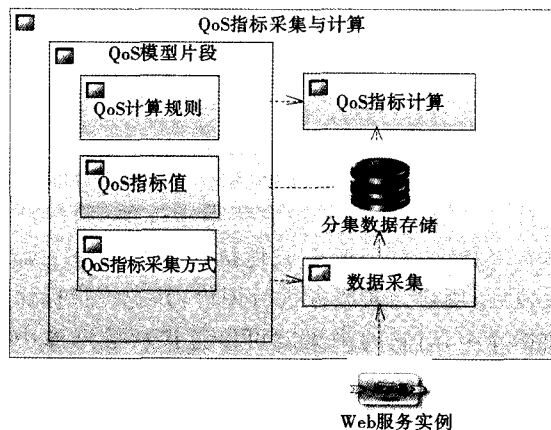


图 4 系统级 QoS 指标采集与计算模块

QoS 模型片段负责从 QoS 模型中获取需要监控的 Web 服务 QoS 指标模型,它包括 QoS 指标计算规则、QoS 指标值和 QoS 指标采集过程三部分。QoS 指标计算模块负责从“采集数据存储设备”中获得采集到的数据,并根据 QoS 模型片段中定义的 Web 服务 QoS 指标计算规则计算出 QoS 指标值。采集数据存储模块负责存储由采集过程采集到的数据和由 QoS 指标计算模块计算得到的 QoS 指标值。

数据采集模块定义了一种采集机制,该机制适用于系统级 QoS 指标数据的采集。它使用了现有的一种开源工具。

图 5 描述了数据采集模块的工作过程。一般情况下,Web 服务的调用分为发送请求、服务响应和服务结束等若干个活动,数据采集模块在相邻的活动之间定义采集插入点,并在采集插入点上嵌入采集扩展模块中定义的采集代码,以采集服务调用与

执行过程中的相关信息(例如:发送请求时间、返回响应时间和 Web 服务的执行状态等),数据采集模块把获得的信息存储到数据库中。

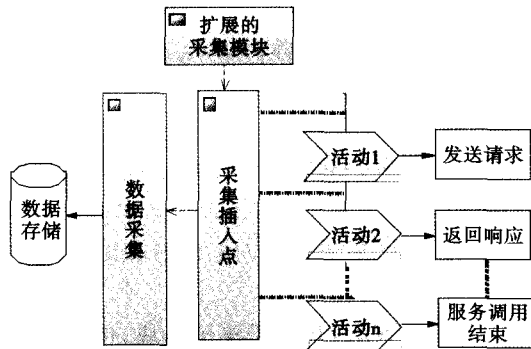


图 5 QoS 信息采集过程图

(2) 供应级 QoS 指标是服务提供者提供的相关服务 QoS 信息. 我们使用 WSDL 描述 Web 服务, 并通过请求/响应模式获得包含供应级 QoS 指标的 SOAP 响应消息。

(3) 消费级 QoS 指标的采集过程比较简单, 可以直接从系统后台数据库中获得指标值. 用户评价是消费级指标. 对于业务流程执行过程中每个服务完成以后, 用户会对服务进行评价, 并把该指标保存在数据库中, 对它的采集就可以直接从数据库中获得。

3.5 上下文信息监控

上下文信息监控模块可以在基于目标建模得到的模型描述文件中获得下文信息。

图 6 描述了用户需求目标模型和它的 XML 描述文件. 左侧是图形化的目标模型, 其中对于 G2 目标经过 OR 分解得到两个子目标 G5 和 G6, 系统运行时会根据上下文信息选择其中一个子目标执行。

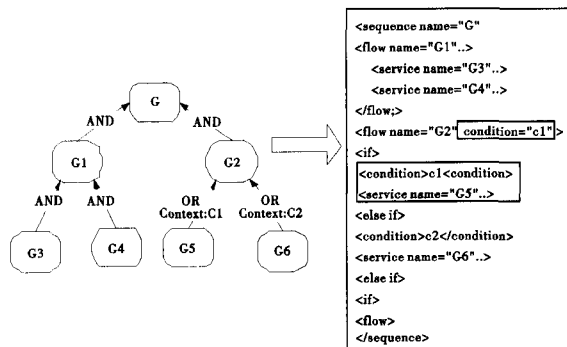


图 6 目标模型和 XML 模型描述

上下文信息 C1 和 C2, 如果 condition="C1", 则执行 G5; 反之则执行 G6. 右侧的 XML 描述文件中, 小方框内定义的是上下文信息以及所选择的子目标。

4 结束语

本文提出了一种基于工作流的监控模型. 已有的监控方法没有完全支持动态的监控, 也就是用户自定义的指标的模型、采集、监控以及计算. 根据用户需求的变化以及环境的变化可以动态的增加指标值的采集、计算、采集等. 本文中只给出了具有代表性的几个 QoS 指标的描述与监控, 在后续的工作中将增加更多常用的 QoS 指标的描述与监控. 提高监控效率, 并且增强监控框架的扩展性。

参考文献:

- [1] Laddaga R. Self-adaptive software[R]. Technical Report 98-12, DARPA Broad Agency Announcement. 1997.
- [2] Salehie M, Tahvildari L. Self-adaptive software: landscape and research challenges[J]. ACM Transactions on Autonomous and Adaptive Systems, 2009, 4(2): 1-42.
- [3] Betty H, Cheng C, de Lemos R, et al. Software engineering for self-adaptive systems: a research roadmap[J]. LNCS, 2009(5525): 1-26.
- [4] Sycara K, Klusch M, Widoff S, et al. Dynamic service matchmaking among agents in open information environments[J]. ACM SIGMOD Record, Special Issue on Semantic Interoperability in Global Information Systems, 1999, 28(1): 47-53.
- [5] Mendiing J, Neumann G, Nuttgens M. Towards workflow pattern support of event-driven process chains (EPC) [C] // Proc. of the 2nd Workshop XML4BPM. Germany: Karlsruhe, 2005.
- [6] Peter Y H, Obbons W J. A process semantics for BPMN[J]. LNCS, 2008, 1(5256): 355-374.
- [7] James S, Thomas H, Axel P. Reduced rule base, QoS metrics, running SWS-E and QoSBroker[R]. European IST Project: FP6511723 Technical Report, 2005.

作者简介:

章小卫 男, (1982—), 硕士, 讲师. 研究方向为智能软件。