动态Web服务行为适配方法初探

曹国荣 谭庆平 吴 浩
(国防科技大学计算机学院 长沙 410073)

摘 要 面向服务的计算是当前软件工程领域和软件产业界的研究热点之一。随着Web服务组合技术的广泛应用,服务间的行为交互日趋复杂,已存在的静态Web服务行为适配方法难以支持复杂Web服务间的行为适配。以此为背景,首先介绍Web服务适配的相关概念,然后分析现有的静态Web服务行为适配方法的研究现状和存在的主要问题,最后探讨动态Web服务行为适配的基本原理、一般方法和技术优势。

关键词 面向服务的计算,服务适配,静态适配,动态适配,适配引擎,适配器
中图法分类号 TP393 **文献标识码** A

Research on Dynamic Web Service Behavior Adaptation

CAO Guo-rong TAN Qing-ping WU Hao
(School of Computer, National University of Defense Technology, Changsha 410073, China)

Abstract Service-oriented computing is a research focus of current software engineering and software industries. With the widespread application of Web service composition, behavior interactions/collaborations among services become increasingly complicated. Current static mechanisms for service behavior adaptation are hardly to support adaptation of interactions/collaborations among complex Web services. Against this background, this paper firstly introduced fundamental concepts of Web service adaptation, and then analyzed the state-of-the-art of, and challenges to current static service behavior adaptation, finally discussed the basic principles, general methods of dynamic Web service behavior adaptation, and its advantage compared to static behavior adaptation mechanisms.

Keywords Service-oriented computing, Service adaptation, Static adaptation, Dynamic adaptation, Adap-tor engine, Adaptor

1 引言

面向服务的计算技术[1](Service-Oriented Computing, SOC)是一种新的计算范型,提出以服务作为构建软件的基本单元,通过服务组合实现快速、低成本、分布式甚至异构环境下的软件应用。目前,Web服务已成为Internet上资源封装的事实标准和SOC的最佳实现框架。然而Web服务的异构性、动态性、分布式和不断演化的特点使得Web服务之间的协同交互变得异常复杂,如何保障Web服务之间的正确交互成为推动SOC技术应用与实施的关键问题。Web服务适配技术为解决这一问题提供了可行的解决方案,能使原本无法正确协同的服务在服务适配器的介入下正确地完成协同工作。随着Web服务技术的发展,特别是Web服务组合技术的应用,服务间的行为交互协同变得越趋复杂,目前存在的静态的服务行为适配机制难以支持复杂服务间的行为适配,研究针对复杂服务行为的动态的、高效的服务行为适配方法对进一步推动SOC技术的发展、实施和应用具有重要意义。

本文第2节介绍Web服务适配相关的基本概念;第3节探讨静态Web服务行为适配的研究现状及存在的主要问题;第4节介绍动态Web服务行为适配的方法与结构组成,给出该方法适配的一般流程,探讨该方法的技术优势;最后总结全文,指出以后的研究工作。

2 概念介绍

2.1 Web服务框架

图1[2]是Web服务的协议框架,它是SOC技术的具体实现框架之一。Web服务框架遵从面向服务的体系结构(Service-Oriented Architecture, SOA),包含服务提供者(Service Provider)、服务请求者(Service Requester)和服务注册机构(Service Registry)。Web服务框架由一套完整开放的协议和技术规范组成,主要涉及传输层(HTTP、FTP和SMTP)、通讯协议层(SOAP)、服务接口描述层(WSDL)、服务层(WS-BPEL)、业务逻辑层(WS-CDL)和服务注册机构(UDDI)等协议和规范。Web服务框架的出现使得应用组件之间的松散耦合替代了紧密耦合,动态的服务替代了静态的组件,平台不相关性取代了平台依赖性,极大地推动了SOC技术的发展。

到稿日期:2012-02-17 返修日期:2012-08-10

曹国荣(1982-),男,博士生,主要研究方向为形式化方法、Web服务行为适配,E-mail:guorong.cao@gmail.com;谭庆平(1965-),男,教授,主要研究方向为分布式软件工程、形式化方法、Web服务行为适配;吴 浩(1973-),男,博士生,主要研究方向为分布式软件工程、软件演化。

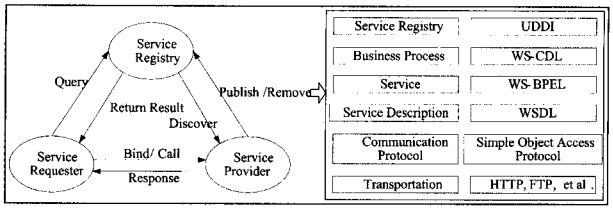

图 1　遵从 SOA 的 Web 服务协议框架

2.2　Web 服务适配

Web 服务适配是指，通过一个适配器，使得一个 Web 服务在经过适配器的封装后能够呈现所期望的服务行为；或者，该适配器作为两个 Web 服务之间的中介，使得原本不兼容的两个 Web 服务能够协调工作。图 2[3] 展示了一个实现服务交互协议替换的适配示例。通过适配器 A 的介入，在不改变用户 Client 与原有服务 S 交互的协议 P 的情况下，将外部交互协议为 PR 的服务 SR 透明地转换为具有服务交互协议 P，如同服务 S 一样的服务，适配器 A 通过协议 PR 和服务 SR 交互，从而使 SR 可以与任何可与服务 S 交互的客户交互。

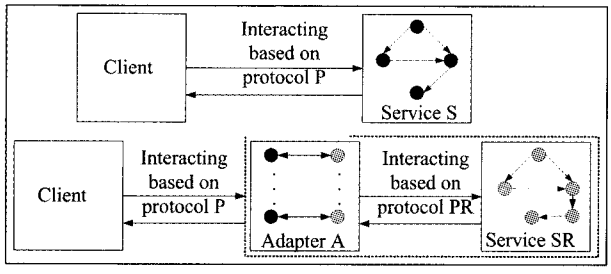

图 2　服务适配示例

根据服务接口模型建模属性的不同，现有的服务适配研究主要包括语法适配、行为适配、语义适配和非功能性适配[4]。语法适配着重从语法上构建服务接口模型，解决服务在方法名、参数名、类型名、复杂类型数据结构、方法类型、参数类型、异常类型、参数顺序和参数个数等不一致情况下的失配问题；行为适配针对服务间的交互行为构建服务接口模型，解决服务在消息交互顺序不一致、消息冗余和消息缺失等情况下出现的失配问题；语义适配将服务接口语义理解为本体，构建基于本体概念的服务接口模型，解决因服务语义描述不一致或者服务功能不完全满足系统需求而引发的失配问题，如概念近似、概念包含和概念等价；非功能性适配，也称为质量适配，基于服务的质量属性构建服务接口模型，解决服务间在安全性、持久性、事务性、可靠性和执行性能等不一致情况下的失配问题。本文关注 Web 服务行为适配方面的研究。

2.3　复杂 Web 服务

根据文献[5]可将 Web 服务分为无状态 Web 服务和有状态 Web 服务。无状态 Web 服务的接口各操作之间相互独立，不存在依赖关系，例如很多信息查询服务都属于无状态 Web 服务；而有状态 Web 服务的接口各操作之间通常存在严格的逻辑和时序关系，对服务的调用也依赖其当前运行状态。Web 服务行为适配就是为解决有状态 Web 服务间的交互行为不一致问题而被提出来的。根据 Web 服务的动态行为特性，又可将 Web 服务分为简单 Web 服务和复杂 Web 服务。简单 Web 服务的接口各操作之间是简单的顺序时序逻辑和依赖关系；而复杂 Web 服务的接口各操作之间往往会出现分支、循环、并发或它们相互嵌套的时序逻辑和依赖关系。从上述表述可以看出复杂 Web 服务是有状态 Web 服务的特例，是具有复杂行为逻辑的有状态 Web 服务。Web 服务组合也主要是指有状态 Web 服务组合。

随着 Web 服务组合技术的发展和广泛应用，一方面，新的组合服务往往需要根据用户需求、使用限制和环境约束进行定制；另一方面，定制的新的组合服务的组件服务往往是一些已有的组合服务。在这种需求下，Web 服务规模愈来愈大，行为特性也往往趋于复杂。但是，需要指出的是组合服务并不一定是复杂 Web 服务，只有当它的接口操作的时序逻辑和依赖关系满足复杂服务描述时才是复杂 Web 服务，从此意义上讲，原子的有状态 Web 服务也很可能是复杂 Web 服务。区别在于，与原子的有状态 Web 服务相比，组合 Web 服务更有可能具备复杂的逻辑行为特性，成为复杂 Web 服务。

3　静态 Web 服务行为适配

3.1　研究现状

为解决 Web 服务行为失配问题，业界已提出许多解决方案。文献[6,7]提出了基于服务执行轨迹的适配方法，使用 YAWL 对服务建模。其中文献[6]从现有服务的执行轨迹出发寻找是否存在轨迹与目标服务的执行轨迹匹配。该方法要求目标服务的功能是现有服务的子功能，否则就无法找到合适的执行轨迹，从而无法适配。文献[7]则把两个服务的执行轨迹合成作为一个新服务的执行轨迹。该方法在适配过程中，如果路径合成失败，则无法分析失败原因。这两种方法都是基于服务执行轨迹的，很难形式化地证明两个服务的交互过程中不会产生死锁问题。文献[8]基于自动机理论提出模式匹配方法。该方法包括两种服务：提供服务和请求服务。模式匹配方法首先确定与提供服务匹配的所有服务所共同遵循的某种模式，然后再确定请求服务是否满足这个模式，如果满足就证明匹配成功。该方法对服务行为做了较多简化，在实际的适配过程中难以应用。同时，文献[9]也提出一种模式匹配方法。不同的是，该方法使用扩展 Petri 网，即开放工作流网[10] 对服务行为建模，描述服务的外部接口。目前该方法主要用来解决 SOA 中的服务发现和发布问题，但同样可应用于服务适配问题。文献[11]也基于模式匹配思想，但仅限于有限状态服务。文献[12]提出一种基于转换规则的适配器生成方法。该方法在理论上可以应用于任何服务适配，但转换规则及服务间接口映射关系需人工完成，工作量大且易出错。文献[13]提出一种确定并消除服务间接口和协议不匹配的方法。该方法首先为不匹配的服务生成不匹配树，然后根据不匹配树生成适配器描述并最终产生适配器[14,15]。文献[16]提出运用模型检验生成适配器的方法。该方法使用 tableau-based 算法来确定适配器是否存在，如果存在则自动生成满足自定义公平性约束的适配器。尽管模型检验生成适配器的方法较易验证正确性，但状态空间爆炸问题仍是该方法的主要缺点。

3.2　目前存在的问题

从上述相关工作可以看出，目前 Web 服务行为适配仍然停留在静态的服务行为分析与适配阶段。静态 Web 服务行为适配研究在取得一定成果的同时，主要还存在以下问题：

(1)服务行为是非循环的，且存在较多的假设限制。现有方法难以描述服务中的循环，有必要采用新的建模技术对服

务行为建模，使模型更加真实地反映实际中的服务行为。

(2)涉及控制流描述和建模较多，涉及数据流很少。控制流和数据流适配是与服务适配同等重要的两个不同方面，而现有的服务行为建模方法仅考虑服务交互的控制流程，有必要采用新的建模技术对服务的数据流和控制流同时建模，发挥数据流在服务建模和适配中的作用。

(3)当服务规模较大时，面对状态空间爆炸问题力不从心。现有方法在分析复杂服务行为时，无法考虑服务执行过程中出现的全部变量值，有必要采用某种方法以有限的服务行为状态描述表示大量的甚至无穷多的状态。

静态 Web 服务行为适配是在系统设计阶段考虑系统几乎所有可能的运行状况，对 Web 服务行为及其交互过程进行建模，并根据模型分析 Web 服务行为交互的失配点，构建适配器；而动态 Web 服务行为适配是在系统运行时动态确定失配点并在线适配。因此，与动态的 Web 服务行为适配相比，现有的静态 Web 服务行为适配方法还存在以下问题。

(1)无法支持 Web 服务的动态演化或更新。静态 Web 服务适配方法是在 Web 服务运行前确定的，无法在运行时动态调整。系统设计时无法考虑到所有可能出现的运行情况，若运行时出现设计时未预料到的情况，就会导致系统运行时出现异常，失效甚至停机；限于所使用的建模技术，无法对 Web 服务所有细节进行描述和建模，如仅仅考虑控制流，缺乏对数据流的描述等，设计者局限于现有的建模技术和自己对系统分析的深度，使得设计时考虑到的系统运行状况更加片面，这进一步加剧了系统运行的不确定性；Web 服务技术以 Internet 上 Web 服务的动态组合来实现用户的即时需求为目标，用户的需求不断变化，系统时刻都处于动态的演化或更新中，如何维持适配器的同步演化与更新，维持系统运行的一致性是静态适配方法无力解决的。

(2)复杂服务适配的复杂度太高、开销太大。对于简单 Web 服务，其仅含有顺序的时序逻辑行为，运行轨迹单一，采用静态适配方法进行分析和适配，过程较为简单；然而随着服务行为趋向复杂，分析并确认所有可能的运行轨迹将是一项繁重的任务，在某些情况下，如含有循环，几乎是不可能的。用户需求、设计者经验和建模技术上的局限性不可能也不允许对复杂 Web 服务协同交互采用静态 Web 服务行为适配方法。

4 动态 Web 服务行为适配

4.1 适配方法与结构组成

动态 Web 服务行为适配的本质是设计并实现一个通用的适配器引擎，这个适配器引擎以要适配的两个服务的行为协议和协议接口映射为输入，动态地创建并管理针对这两个服务的适配器实例，并在服务协作完成后自动销毁相应的适配器实例。因此，动态 Web 服务行为适配方法的实现包括适配器引擎和适配器框架模型两部分，如图 3 所示。适配器引擎的核心功能就是创建并管理适配器实例，也就是动态 Web 服务行为适配的整个运行生命周期，包括适配器实例的创建及初始化、借助适配器实例实现的动态适配监听处理以及动态服务行为适配结束检测。适配器框架模型是适配器实例的抽象描述，它由要适配的两个服务的行为协议和接口映射构成，在运行时刻由服务通信驱动的具有输入和输出的离散系统模型具体可以用有限状态自动机或进程代数等形式化方法来描述和表示。

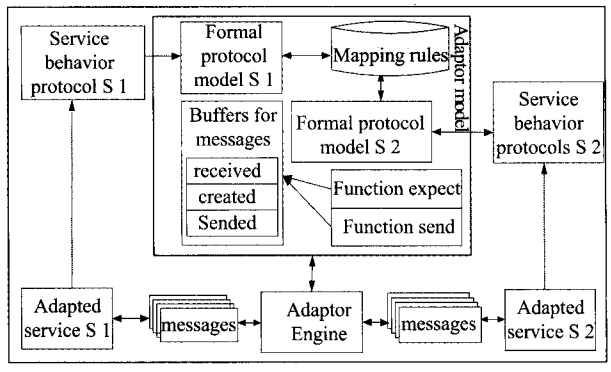

图 3 动态 Web 服务行为适配组成结构

所谓 Web 服务行为协议也称为 Web 服务行为接口，是指服务对外交互过程中实际存在的消息发送和接收等的控制依赖和时序依赖关系，非形式化的相关描述包括 WS-BPEL、WS-CDL 等，形式化的相关描述包括有限状态机、进程代数、Petri 网等。具体采用何种描述手段要根据系统设计需求来选择，如采用有限状态自动机。此时，采用其它方法表示的 Web 服务行为协议必须转换为由等价的有限状态机来描述。

协议接口映射是服务适配的重要组成部分，它描述服务交互双方接口和参数等的映射和转换关系。协议接口映射需要通过对服务交互双方的行为协议进行解析方可获取，目前绝大部分仍需人工解析获取，少部分借助语义 Web 来实现 Web 服务行为协议的自动解析和获取。目前来看，借助语义 Web 实现的服务行为协议自动解析是实现 Web 服务全自动适配的必然选择。

由于 Web 服务适配器框架模型包含了需要适配的 Web 服务行为协议，因此，关于适配器的状态迁移就可以通过两个要适配的 Web 服务各自的状态迁移组合表示。此外，Web 服务适配器框架模型还需要一些必要的内部结构和机制来支持适配器引擎的正常运行和适配。这包括：(1)Web 服务交互和适配过程中接收消息、产生消息和发送消息的缓存机制；(2)适配器框架模型应能够根据当前适配器状态计算可接收的消息集合、需要发送的消息集合等。

4.2 动态 Web 服务行为适配流程

一个完整的动态 Web 服务行为适配流程应包括 3 个阶段：适配器创建及初始化阶段、监听处理阶段(包括消息接收、映射选取、消息转换和消息发送等主要步骤)和适配结束检测阶段。

适配器创建及初始化阶段。在该阶段，适配器引擎获取要适配的两个服务的行为协议并将其转换为适配器框架模型所识别的形式化表示，然后和协议接口映射一起作为适配器创建后初始化的输入参数完成适配器初始化。

适配器完成初始化后，适配器引擎进入监听处理阶段，如图 4 所示。此阶段又可分 4 步进行：

1)消息接收。引擎在接收到消息后，立即根据适配器当前状态，调用 function expect 判别接收到的消息是否需要适配，若不是，则引擎回到监听初始阶段并继续监听；否则引擎将该消息存于消息缓存 received 中，然后刷新适配器到新的状态，并进入步骤 4)。

2)映射选取。引擎根据适配器当前状态，调用 function

send 获取要在当前状态下发送的消息的类型；然后遍历其中的每一个消息类型并与映射匹配，查看适配器缓存 created 和 received 中的消息是否满足映射要求，若满足，则进入步骤 3)；该步骤一直持续到没有任何消息可满足映射要求为止，此时进入步骤 1)。

3)消息转换。若缓存 created 和 received 中的消息满足映射要求，则根据转换规则，调用转换函数生成对应的目标消息，该目标消息保存于缓存 created 中，然后进入步骤 4)。

4)消息发送。遍历缓存 received 以及 created 中的消息是否属于需要发送的消息类型，若是，则发送该消息，同时将该消息保存于缓存 sended 中。遍历完成以后进入步骤 2)。

适配结束检测阶段。当适配器引擎处于监听阶段时，此判定过程同步执行，计算适配器当前状态下是否还存在要接收的消息类型，若没有，则适配过程结束，这标志着要适配的两个服务已在适配器协助下完成服务协同；否则循环继续该检测过程。

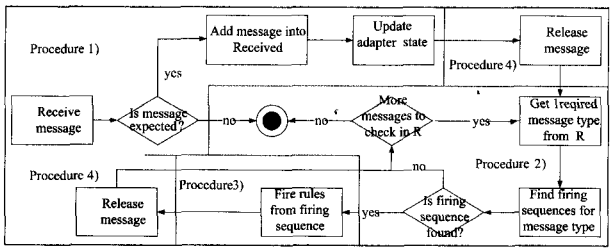

图 4　动态 Web 服务行为适配监听处理阶段流程图

4.3　优越性

与静态 Web 服务行为适配方法相比，动态 Web 服务行为适配方法具有许多优势：

首先，无须在设计系统时进行复杂的服务行为兼容性分析和适配，能有效降低软件设计成本。无论是静态方法还是动态方法，都需要以适配器为中心完成消息的接收，处理与转发都需要以服务行为接口源消息和对应的目标消息为映射对建立映射，将其作为构建适配器的输入信息。适配方法根据交互双方行为协议描述和映射等输入信息生成静态适配器，或动态地接收源消息，并通过查找映射库找出对应的目标消息并转发完成动态适配。因此，动态适配方法将设计阶段的服务行为兼容性分析和适配推迟到运行阶段，由第三方(适配器引擎)在服务运行时完成服务交互行为的兼容性分析、映射匹配和行为适配。

其次，无须产生不兼容服务的完全适配器，具有统一的适配框架机制。静态 Web 服务行为适配方法需要针对被适配的服务双方生成唯一的适配器，不同的被适配的 Web 服务间具有完全不同的适配器，适配器是针对特定服务双方的。也就是说如果需要 $2n$ 个服务协同实现某项任务，且假设在极端情况下，这些两两交互的服务间都存在失配问题，那么最终需要为每对服务生成对应的适配器，这样就需要产生 n^2 个不同的适配器，在系统规模较大时，适配器过多，根本无法维护。动态 Web 服务行为适配方法则从 Web 服务实例交互行为细粒度上实现服务行为适配，该机制无须对服务双方进行整体行为兼容性分析并产生全局适配器，对于 $2n$ 个服务的协同适配问题，用统一的适配框架来保证它们实现正确的协同交互，也就是说这些服务具有统一的适配器。与上述适配器的不同在于该适配器能够感知上下文，动态接收、处理和转发服务交互信息，是一个智能的自治实体，虽然其与上述适配器相比复杂得多，但是它使适配器的数量急剧下降，系统设计工作量大大降低。当然，与静态服务行为适配方法相比，如何提高适配器执行的性能仍然是动态服务行为适配方法的焦点和难点。

最后，系统易于扩展和维护，在动态演化或升级时，具有很强的伸缩性和适应性。由于动态适配采用统一的适配框架而无须维护数量众多的适配器，因此系统变得易于扩展和维护。统一的适配框架无须添加或修改某个特定适配器，在 Web 服务动态演化和在线升级时，只须更新适配框架的服务描述和映射匹配信息，因此具有很强的灵活性和适应性。

结束语　本文针对静态 Web 服务行为适配方法无力解决复杂 Web 服务行为适配这一现状，给出了一个动态 Web 服务行为适配的框架模型和一般适配流程，结合实例分析，探讨了动态 Web 服务行为适配的优越性，为解决复杂 Web 服务行为适配提供了新的思路。

随着 Web 服务组合技术的进一步发展，如何保障复杂 Web 服务行为之间的正确交互成为 Web 服务行为适配研究新的关注点。目前国外在该领域已经取得部分成果[17]，该方面的研究内容包括：(1)更精确的 Web 服务行为形式化模型。不仅对服务行为的控制流建模，还对数据流、状态变迁和消息传递等动态行为属性进行建模，以实现对服务交互行为进行严谨的分析和推导。(2)针对复杂 Web 服务交互行为的高效的服务自动适配机制。从 Web 服务运行实例行为细粒度上动态地分析 Web 服务间的交互正确性并识别失配类型，运行时根据映射规则实现适配。(3)形式化 Web 服务行为适配正确性验证和分析。对服务交互的正确性进行分析与验证，特别是涉及数据流、循环服务行为时，如何验证 Web 服务交互的正确性，分析各种失配行为。

未来的研究工作将围绕静态 Web 服务行为适配机制在支持复杂服务行为协同交互的适配与验证方面存在的关键技术问题，重点研究动态 Web 服务行为适配和适配正确性验证方法。目前，在 SOC 领域，高效、动态的服务行为自动适配方法是公认的难题，如果能取得预期的成果，将会进一步推动 SOC 技术的发展、实施和应用。

参 考 文 献

[1] Papazoglou M P, Traverso P, Dustdar S, et al. Service-Oriented Computing: A Research Roadmap [J]. International Journal of Cooperative Information Systems, 2008, 17(2): 223-255

[2] 陈振邦. 服务计算中接口模型与构件设计的研究[D]. 长沙：国防科学技术大学计算机学院，2009

[3] Benatallah B, Casati F, Grigori D, et al. Developing Adapters for Web Services Integration [C]// Proc of the 17th International Conference on Advanced Information Systems Engineering. 2005: 415-429

[4] Becker S, Brogi A, Gorton I, et al. Towards An Engineering Approach to Component Adaptation [C]// Proc of the 2004 International Dagstuhl Seminar on Architecting Systems with Trustworthy Components. 2004: 193-215

[5] 杨艳萍. 自动 Web 服务组合关键技术研究[D]. 长沙：国防科学技术大学计算机学院，2007

[6] Brogi A, Popescu R. Service Adaptation through Trace Inspection [C]// Proc of SOBPI'05. 2005: 44-58

(下转第 29 页)

menta Informaticae,2011,108:267-286

[3] Li H X,Zhou X Z. Risk decision making based on decision-theoretic rough set:a three-way view decision model [J]. International Journal of Computational Intelligence Systems, 2011, 4 (1):1-11

[4] Li H X,Zhou X Z,Zhao J B. Positive region-based reduction in decision-theoretic rough set model [J]. Transaction on Rough Sets

[5] Liu D, Li T R, Ruan D. Probabilistic model criteria with decision-theoretic rough sets [J]. Information Sciences, 2011, 181 (17):3709-3722

[6] Liu D, Li T R, Li H X. A multiple-category classification approach with decision-theoretic rough sets[J]. Fundamenta Informaticae,2012,115(2/3):173-188

[7] Liu D,Li H X,Zhou X Z. Two decades' research on decision-theoretic rough sets [C]//Proceeding of 9th IEEE International Conference on Cognitive Informatics. 2010:968-973

[8] Liu D,Yao Y Y,Li T R. Three-way investment decisions with decision-theoretic rough sets [J]. International Journal of Computational Intelligence Systems,2011,4(1):66-74

[9] Ma W M,Sun B Z. On relationship between probabilistic rough set and Bayesian risk decision over two universes[J]. International Journal of General Systems,2012,41(3):225-245

[10] Mishra A,Mishra M,Shiv B. In praise of vagueness:malleability of vague information as a performance -booster [J]. Psychological Science,DOI:10. 1177/0956797611407208

[11] Pawlak Z. Rough sets [J]. International Journal of Computer and Information Sciences,1982(11):341-356

[12] Pawlak Z,Wong S K M,Ziarko W. Rough sets:probabilistic versus deterministic approach [J]. International Journal of Man-Machine Studies,1988,29:81-95

[13] Pawlak Z,Skowron A. Rough membership functions [M]. Advances in the Dempster-Shafer Theory of Evidence. John Wiley and Sons,New York,1994:251-271

[14] Wong S K M, Ziarko W. Comparison of the probabilistic approximate classification and the fuzzy set model [J]. Fuzzy Sets and Systems,1987,21:357-362

[15] Yang X P,Yao J T. Modeling multi-agent three-way decisions with Decision-theoretic rough sets [J]. Fundamenta Informaticae,2012,115:157-171

[16] Yao Y Y,Wong S K M. A decision theoretic framework for approximating concepts [J]. International Journal of Man-machine Studies,1992,37:793-809

[17] Yao Y Y, Zhao Y. Attribute reduction in decision-theoretic rough set models [J]. Information sciences, 2008, 178: 3356-3373

[18] Yao Y Y. Three-way decisions with probabilistic rough sets [J]. Information Sciences,2010,180:341-353

[19] Yao Y Y. The superiority of three-way decision in Probabilistic rough set models [J]. Information Sciences, 2011, 181: 1080-1096

[20] Zadeh L A. Fuzzy sets [J]. Information and Control, 1965,8 (3):338-353

[21] 顾婧,周宗放. 基于可变精度粗糙集的新兴技术企业信用风险识别[J]. 管理工程学报,2010,24(1):70-76

[22] 贾修一,商琳,陈家骏. 决策风险最小化属性约简[J]. 计算机科学与探索,2011,5(2):155-160

[23] 李华雄,刘盾,周献中. 决策粗糙集模型研究综述[J]. 重庆邮电大学学报:自然版,2010,22(5):624-630

[24] 李华雄,周献中,李天瑞,等. 决策粗糙集理论及其研究进展[M]. 北京:科学出版社,2011

[25] 刘盾,姚一豫,李天瑞. 三枝决策粗糙集[J]. 计算机科学,2011,38(1):246-250

[26] 刘盾,李天瑞,李华雄. 区间决策粗糙集[J]. 计算机科学,2012,39(7):178-181,214

(上接第 13 页)

[7] Brogi A,Popescu R. Automated Generation of BPEL Adaptors [C]//Proc of the 4th International Conference on Service-oriented Computing. 2006:27-39

[8] Massuthe P, Wolf K. An Algorithm for Matching Non-deterministic Services with Operating Guidelines [J]. International Journal of Business Process Integration and Management, 2007, 2(2): 81-90

[9] Massuthe P,Reisig W,Schmidt K. An Operating Guideline Approach to the SOA [M]. Annals of Mathematics,Computing & Teleinformatics,2005:35-43

[10] Schmidt K. Controllability of Open Workflow Nets [C]//Enterprise Modelling and Information Systems Architectures. 2005: 236-249

[11] Lohmann N,Massuthe P,Wolf K. Operating Guidelines for Finite-State Services [C]//Petri Nets and Other Models of Concurrency-ICATPN 2007. 2007:321-341

[12] Gierds C, Mooij A J, Wolf K. Specifying and Generating Behavioral Service Adaptor based on Transformation Rules [R]. Universität Rostock,Germany,2008

[13] Nezhad H R M,Benatallah B,Martens A,et al. Semi-Automated Adaptation of Service Interactions [C]//16th World Wide Web Conference. ACM Press,2007:993-1002

[14] Bracciali A,Brogi A,Canal C. A formal Approach to Component Adaptation [J]. Journal of Systems and Software,2005,74(1): 45-54

[15] Brogi A, Canal C, Pimentel E. On The Semantics of Software Adaptation [J]. Science of Computer Programming, 2006, 61 (2):136-151

[16] Sinha R,Roop P,Basu S. A Model Checking Approach to Protocol Conversion [J]. Electronic Notes in Theoretical Computer Science,2008,203(4):81-94

[17] Kenneth W,Marlon D,Chun O Y. The Service Adaptation Machine [C]//Proc of the 6th European Conference on Web Services. 2008:145-154