基于面向服务体系结构的遗留系统集成方法研究*)

熊安萍　王化晶　瞿　中
(重庆邮电大学计算机科学与技术学院　重庆 400065)

摘　要　面向服务的体系结构是为解决开放环境下业务集成的需要、通过连接完成特定任务的功能实体加以实现的一种软件系统架构。本文针对组织内现存的各种遗留系统，通过将其包装成"服务"的思想，提出了一种遗留系统集成方法，并结合实例给出了它的体系结构和实现框架，能够较好地解决遗留系统互操作及重用问题。

关键词　体系结构，面向服务，遗留系统，集成方法

Research on Integration Method of Legacy Systems Based on Service-oriented Architecture

XIONG An-Ping　WANG Hua-Jing　QU Zhong
(College of Computer Science & Technology, Chongqing University of Posts & Telecommunications, Chongqing 400065)

Abstract　Service-oriented architecture is a software system architecture which addresses the needs of the business integration in open environment. It does so through interconnecting functional entities that implement specific task. By analyzing the various legacy systems, a method of legacy systems integration is presented according to the idea that packs them as services. Moreover, an example is given of architecture and implementation of the framework. This solution is able to leverage the issue of interoperability and reuse of legacy systems.

Keywords　Architecture, Service-oriented, Legacy systems, Integration method

1　引言

近几年，面向服务的体系结构(Service-Oriented Architecture，SOA)[1,2]迅速兴起。SOA是为了解决在Internet环境下业务集成的需要，通过连接完成特定任务的功能实体实现的一种软件系统架构。它强调的"重用"和"互操作"特性，使快速开发、集成和重用应用成为可能。同时，基于SOA架构的系统能在业务发生变化后动态响应新需求，快速重新编排各种软件构件和服务生成新的应用。

遗留系统(Legacy System)是指任何作为产品在组织内运行的系统，一般由效能较低的语言实现，运行的硬件环境落后且平台相关，维护与扩展困难。这些系统由于在开发时未考虑互操作及集成需求，其间不能很好地协同运作与资源共享。但遗留系统往往承载了组织的关键应用，不能简单丢弃，所以最大限度地重用遗留系统一直是产业界的研究热点。本文通过将遗留系统(或其组件)包装成"服务"，提出了一种基于SOA架构的遗留系统集成方法，较好地解决了其重用和互操作问题。

2　面向服务的体系结构

SOA是一种粗粒度、松散耦合的服务架构。W3C将服务定义为："服务提供者完成一组工作，为服务使用者交付所需的最终结果。"在SOA中，服务是软件的基本组成单元，服务之间可以相互通信，并且具有自治性、平台无关性及灵活性。

服务可通过3种途径获得：从零开始开发服务、将遗留系统包装成服务或从市场上采购COTS(component off-the-shelf，商用成品组件)。借助SOA服务动态组合的特征，可以构建灵活、可伸缩、敏捷应对变化的应用系统，如图1所示。

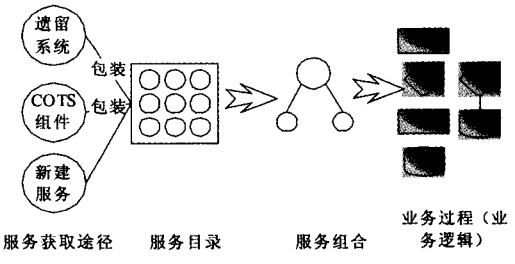

图1　服务组合示意图

SOA作为一套"设计哲学"独立于任何特定技术，目前大多基于Web Service方式实现。Web Service使用一系列标准和协议实现相关功能，如使用WSDL(Web Service Definition Language，Web Service描述语言)描述服务，使用UDDI(Universal Description, Discovery, and Integration，统一描述、发现和集成)发布和查找服务，使用SOAP协议(Simple Object Access Protocol，简单对象访问协议)执行服务调用等，其结构模型如图2所示。

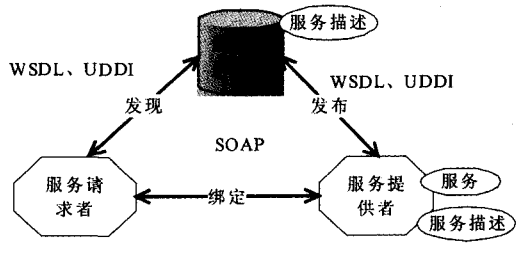

图2　基于Web Service的SOA体系结构模型

*)本研究由重庆市科委自然科学基金计划资助项目(No. CSTC2007BB2454)、重庆市信息产业发展政策研究重点项目(No. K2007-49)和重庆市高等教育教学改革立项重点项目(No. 0624057)资助。熊安萍　副教授，主要研究方向为软件工程、计算机网络与通信、操作系统内核等；王化晶　硕士研究生，高级工程师，主要研究方向为数据与应用集成、软件工程、电子政务等；瞿　中　主要研究方向为计算机网络与通信。

为发挥SOA在应用与数据集成领域的潜能，人们提出了ESB(Enterprise Service Bus，企业服务总线)的概念。ESB是一个基于开放标准的通信骨干，用于消除诸多平台及具体应用间的异构性，促进平台间的互操作和不同格式数据间的共享。

因此，SOA是一个所有功能均被分解成精确定义的、可调用的、相对独立的服务，且服务能被有序编排构建业务流程的应用架构，它满足遗留系统集成的需求。

3 集成框架

SOA是通过在遗留系统上增加包装器实现集成的。包装器为原有系统增加服务接口，所有接口根据SOA协议提供统一的服务定义和调用方式，而忽略具体的实现细节。其具体实现则由各遗留系统提供，服务接口仅负责数据的格式、类型等转换工作。通过将遗留系统中需要暴露的功能包装成Web Service，遗留系统之间既可通过Web Service进行信息交互，又保证了各自功能、安全稳定性不受影响。图3给出了遗留系统的集成框架。

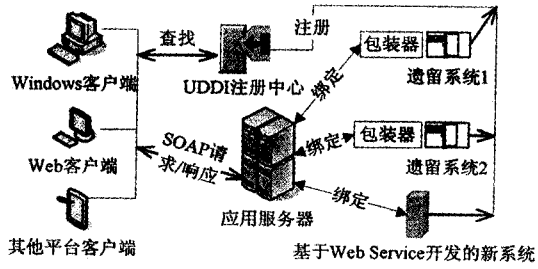

图3 遗留系统集成框架

该框架具有4个特点：①SOAP利用XML描述系统间的交互信息，同时使用Http和SMTP使遗留系统能够跨平台集成；②Web Service提供完全的封装性，实现遗留系统功能的完整封装，且不影响其安全稳定性；③可根据集成策略的变化重新集成各遗留子系统，保证集成的动态性；④可实现遗留系统的多入口集成，保证集成策略的灵活性。

4 集成的关键问题

集成遗留系统需要解决3个方面的问题[3]：①语义管理。许多遗留系统有内部私有数据，它们为满足所属系统需求而定义，设计时并未考虑日后与其它数据源间的交换和互操作，这些数据只能被自己对应的系统解释和处理，故消除这种语义异构性是首先要考虑的问题。②服务标识、划分和实现。通常并不是遗留系统中所有业务逻辑和功能都对外开放，因而要标识出哪些部件(或功能)需要被包装成服务。同时，每个服务的粒度要仔细斟酌，因为粒度大小会影响业务的灵活性和实现服务的复杂性。③性能影响。松散耦合系统的性能必然低于紧密耦合系统，原因有3个：一是SOAP基于XML文本，系统在传输和处理数据时必须进行二进制流到文本之间的转换；二是Web Service面向Internet，服务器和客户端的通信带宽受到一定限制会产生延时；三是用于提供端到端安全的WS-Security和其它相关标准也会在一定程度上增加系统负载。

4.1 语义管理

集成遗留系统，首先要在不同系统间实现语义互操作。目前，较理想的方式是利用XML(Extensible Markup Language，可扩展标记语言)作为数据表示和交换的标准，通过一定的映射规则，达到无缝集成的目的。在语义映射及XML语义集成方面已有大量研究及实现方法，如文[4,5]。

私有数据与XML间的数据转换主要有直接转换与间接转换两种方法：直接转换是指构造专用转换逻辑，每进行一次数据转换就执行一次转换逻辑，适用于不需要进行数据的本地存储、数据量小或转换逻辑不很复杂的情形。间接转换是指分配缓存作为存储中介，数据仅与存储中介交互而不直接转换，其适用条件与直接转换相反。

4.2 服务标识、划分和实现

遗留系统中需要暴露的功能部件和数据单元应进行分析和标识，以确定哪一部分功能、哪些组件或数据源要当成服务处理。遗留系统的功能通常包括进行业务逻辑处理或仅提供结果。若业务逻辑需要不断地重复使用，可考虑当作服务；若仅对外提供执行结果，直接包装成数据服务即可，一般称为IaaS(Information as a Service)或DaaS(Data as a Service)[6,7]。服务标识过程中应坚持两个原则：①命名服务时以最大化易用性为目标，应使用专业领域内有意义的名称并优先选用业务概念，用名词对服务命名，用动词对操作命名；②被标识的服务应是内聚且完整的。

服务划分的主要任务是确定服务粒度的大小、定义服务的层级结构及包含关系。服务粒度太大不利于组合，且使系统灵活性差；粒度太小又不便于软件部署，并增大系统开发的复杂性。在确定服务粒度时，需要系统分析人员在多个因素间进行折衷，如可维护性、可操作性、易用性和可组合性等。Lewis等人提出了面向服务的迁移和重用技术[8](Service-Oriented Migration and Reuse Technique，SMART)，为服务封装评估提供了系统化的方法和工作流程。

遗留系统作为服务有两种实现方式，即作为服务提供者或服务消费者。作为服务提供者：遗留系统整体(或部分组件)当作SOA端点(end point)[9]。处理流程是：①遗留系统收到SOAP消息；②处理消息，取出包含对方数据的XML载荷；③XML数据被转换成遗留系统的私有格式，并递交给相应组件；④处理请求，产生反馈信息并将其转换成XML格式；⑤根据XML载荷数据生成SOAP消息并回送，如图4所示。

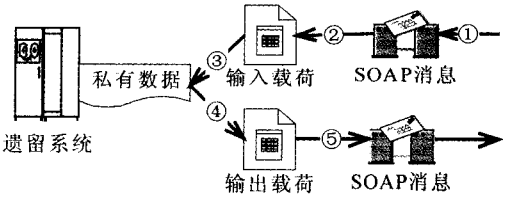

图4 遗留系统作为服务提供者的处理流程

作为服务消费者：遗留系统需要提供调用能力，以便和与其交互的其它服务进行通信。处理流程如下：①遗留系统准备发出一个请求时，首先将私有数据转换成XML格式。如果该请求与私有数据无关，此步骤可省；②生成封装有XML数据的SOAP消息并发出；③对方收到消息，调用相应服务；④反馈消息回送给遗留系统；⑤处理SOAP消息，获得需要的反馈信息。

遗留系统中标识出服务并增加接口(包装器)后，可将其在服务注册中心发布。当调用服务时，其它服务可通过UDDI协议在服务注册中心查找相应服务，获得其地址和调用参

数。根据获得的地址发起连接，然后通过远程对象绑定进行请求的发布和应答的接收，从而实现交互。

4.3 性能优化

解析 SOAP 消息、XML 和私有数据间转换均要增加处理时间，因此会在很大程度上影响系统性能。XML 处理时间取决于数据量大小和结构复杂性，数据量越大，结构越复杂，处理时间越长，于是减少交互数据量和简化数据结构是两个应优先考虑的手段。另外，如果客户端在发出 Web Service 调用请求后不要求立即得到结果，可以运用异步调用方法，在发出异步调用请求后转到其它处理上，等 Web Service 返回结果后再回到先前的处理过程。

另外，还要考虑使用 Web Service 传输二进制数据的性能。Web Services 发送/接受二进制数据通过 3 种方式进行：①Base64 方式，它简单易用，但生成的文本对应二进制数据体积有 30%以上的膨胀，发送很少数据时可用。②带附件的 SOAP，这种做法有效而标准，不过附件只是数据的 URL，对端收到消息后通过 URL 得到数据。这容易在互操作性上出现问题，如和 WS-Security 协作时可能控制不到外部的二进制数据。③Message Transmission Optimization Mechanism (MTOM)，它通过 XML-binary Optimized Packaging (XOP) 方式实现二进制数据传输。MTOM 也将数据放在 XML 文档外面，但通过在 XML 文档中添加 XOP：include 元素告之 XML 处理器使用二进制数据替换特定的内容，这使得二进制数据处理方式与文本一致。因此，如果传送一个大的二进制数据文件不可避免，则应尽可能采用后两种方式，其中 MTOM 效率较高。

5 应用实例

上述方法已应用于突发公共事件应急响应信息系统(Emergency Response Information Systems，ERIS)的开发过程中。ERIS 是应对突发公共事件的专用信息系统，目标是实现用户间的任务协同，提高即时决策效率和水平[10]，它对应急响应组织间通信、数据收集和分析、制定决策等提供支持。本文提出的设计思路是将目前已存在的各类应急信息系统，如疾病防控系统、道路监控系统、防洪测报系统等作为遗留系统对待，通过对其的适当包装统一到 SOA 框架之下。

理想的 ERIS 应具有如下 3 个基本特征：基于分布式体系结构[11]；各分布结点能够协同运作；系统动态构建(improvise)[12]。因此，ERIS 应该能动态、灵活地配置和部署，以适应突发事件现场形势发展的应对需要。

5.1 系统体系结构

基于 SOA 架构的 ERIS 在逻辑上划分为 4 个层次：表示层、流程编排层、服务接口层和基础设施层，如图 5 所示。

(1)表示层作为整个系统的门户，一般通过 Web 方式展现，利用 GUI(图形用户界面)为用户提供统一的信息服务功能入口。表示层将内部和外部各种相对分散独立的信息组成统一的整体，保证了系统用户既能够从一致的渠道访问其所需信息，也可以依据每一个用户的要求提供个性化服务。

(2)流程编排层根据突发公共事件的进展及当前情势，动态、灵活地组合需要的服务，将不同的应用系统连接在一起，进行协同工作，并提供流程管理的相关功能，如流程设计、监控和规划等。

(3)服务接口层要解决应急指挥中心与各专用应急系统之间的连接和数据接口问题，即服务包装和服务组合的具体实现，主要功能是将遗留系统包装成服务。同时，它也为各类服务的存储、查找提供目录服务，完成数据的统一转换和发送，以及消息路由和消息队列管理等。

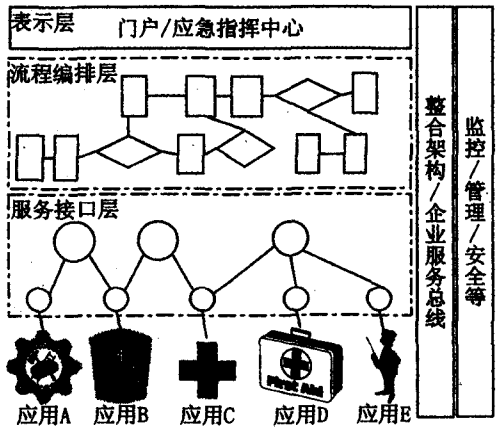

图 5 应急响应信息系统体系结构

(4)基础设施层主要包括网络及通讯设施、服务器及存储系统、视频会议系统、大屏幕显示系统等，这是系统的运行基础。考虑到 ERIS 的实施主体均为各级党政机关，系统依托电子政务内网，采用 VPN 技术(Visual Private Network，虚拟专用网)作为 ERIS 的底层网络环境。

5.2 系统实现方式

图 5 所示体系结构的实现可以采用 J2EE 和. Net 两种技术。本系统以 J2EE 实现，其运行时框架如图 6 所示。表示层通过组合 HTML、JSP 和 Java Servlets 技术实现，服务接口层利用 J2EE 提供的组件和服务(EJB)实现，流程编排层用 BPEL 实现。BPEL4WS(Business Process Execution Language for Web Service)是一种抽象程度较高的流程描述语言，其整个规范中没有出现任何与底层网络协议相关的部分，所有的信息传输都基于 SOAP 协议完成。BPEL4WS 最大的特点是它没有使用专有的语言规范，而是基于已广为流传的 XML 来创建自己的流程。

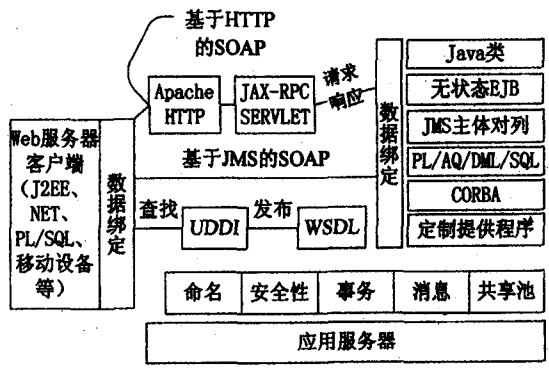

图 6 基于 J2EE 的 SOA 运行时框架

结束语 SOA 架构可以实现网络环境下应用的松散耦合和动态集成，能使用户方便快捷地集成现有的应用和部署新应用。本文基于 SOA 架构，提出了一种遗留系统的集成方法，并结合实例给出了其体系结构和实现框架。实践证明，上述方法能较好地解决遗留系统的动态集成与部署，提高了遗留系统的重用水平。

(下转第 282 页)

的结构信息和实施动作的所有回调函数等),而应用类(如在例子中的 DataAcq 类) 作为动作的上下文,即应用类为行为类提供功能参数包括特殊用途的变量。这种方法需要应用类显式地存储状态配置信息,因为行为类是无状态的。

表 1 dynamic_behavior 包中实施模式对应的 Java 类

EHA 类	-collectEnabled(Event, Configuation Object) : Transition[] -collectFireable(Transition[]) : Transition[] -RestartTransition(Configuration Object, Transition) : void +dispatchEvent(Event, Configuration Object) : void +initialize(Object) : Configuration
Transition 类	triggerEventClass: Class aasociatedAction(Object) : void guard(Object) : bool
Configuration 类	isActive(State) : bool markActive(State) : void markinactive(State) : void
State 类	entryAction(Object) : void exitAction(Object) : void
Automaton 类	……
Event 类	……

实现事件处理引擎的模式必须包括两部分的转化:基本的支持类是在动态包内,而自动产生的类则是在生成包里,如表 1 所示。

从结构化观点看,如何在 Java 中表示 EHA、自动机、状态、迁移和事件类,上述的类都是 EHA 元模型的对应 Java 等值体,它们的实例及之间的关联都是元模型在 Java 语言上的映射。

从如何插入 Java 编码安全动作等动态应用的观点上看,这些类方法是抽象概念上的点(actions, guards 等),最后被具体编程语言级别上的实现所替代:状态入口和出口动作是状态类的方法(State. entryAction, State. exitAction),而动作与变迁相关联,关口是变迁类的方法(Transition. associatedAction, Transition. guard)。因为动作和关口是对具体状态和变迁的定义,这些类是抽象的,它们的方法将在派生类中实现。当前状态配置由与当前活动状态相关联的配置类表达。所有基于 EHA 行为的活动状态都应该包含在这个类的一个实例中。

转化由 EHA 类实现,dispatchEvent 和 initialize 方法是入口。如同上面讨论到的,活动应用类充当方法的普通参数。initialize 方法用于初始化活动对象的配置。真正的事件派发由 dispatchEvent 功能实现。这个函数作为功能参数被调用,由应用类负责传递事件、配置和自身引用。dispatchEvent 方法首先调用 collectEnabled 方法,并透过 Transitions 实例的关联(充当 allTransitions 角色)收集变迁,然后通过调用 collectFireable 方法从这个集合中去除由优先权关联造成的失活变迁(Transition. disabling 关联)。在执行对 RestartTransition 方法相应的调用期间,因掉电重起而选择的变迁将被执行;这里包括(1) 进行状态退出动作(即调用 State. exit 方法);(2) 进行动作关联变迁;(3) 进行状态入口动作(即调用 State. entry);(4) 更新配置。

在编程人员实现的方法中捕捉可能发生的异常并将其转换为异常事件。将 Java 语言定义的结构(即在 Java 中的 try-catch 块)用于整合表 1 中的功能,转化方法不需要任何涉及到实际行为模式的修改,真正重要和易出错的部分,如建立对象结构表达 EHA 等,由编码发生器自动创建。

结论 本文根据 EHA 结构化地描述 UML Statecharts 及其操作语义,将 Java 异常处理 Statecharts 映射为 EHA 自动机,然后基于自动机理论的模型检验方法以类似 Java 语言异常处理的方式(try-catch-finally 结构)对可能发生的异常能够做出响应,产生相应的代码。由于本文的提出方法不要求对 UML Statecharts 语义做修改,因此遗留模型检验甚至在异常出现时都可用于检查系统的行为,为基于异常感知的状态图源代码自动发生器的实现提供了一个良好的基础。

参 考 文 献

1 Object Management Group. Unified Modeling Language Document Set. Version2.0, OMG document formal /05-07-04. 2005
2 Harel D. Statecharts: a visual formalism for complex system. Science of Computer Programming, 1987, 8 (3) : 231～274
3 Mikk E, Lakhnech Y, Petersohn C. On formal semantics of statecharts as supported by STATEMATE. In: The 2nd BCS-FACS Northern Formal Methods Work shop. Spring-Verlag, 1997
4 Petersohn C. Data and control flow diagram s, statecharts and Z: their formalization, integration and real-time extension [Ph. D. Thesis]. Germany, 1997
5 Latella D, Majzik I, Massink M. Towards a formal operational semantics of UML Statechart diagrams. In: Ciancarini P, Fantechi A, Gorrieri R, eds. Proceedings of the 3rd International Conference on Formal Methods for Open Object-Oriented Distributed Systems. Boston: Kluwer Academic Publishers, 1999. 15～18
6 董威,王戟,齐治昌. UML Statecharts 的模型检验方法[J]. 软件学报, 2003, 14 (4) : 750～755
7 VarróD, VarróG, Patericza A. Checking General Safety Criteria on UML Statecharts. In Lecture Notes in Computer Science, number 2187. Springer Verlag, 2003. 46～55

(上接第 275 页)

参 考 文 献

1 Papazoglou M P, Heuvel W. Service oriented architectures: approaches, technologies and research issues[J]. Very Large Database Journal, 2007, 16(3): 389～415
2 Erl T. SOA Principles of Service Design[M]. Prentice Hall, 2007. 25～66
3 Canfora G, Fasolino A R, Frattolillo G, et al. Migrating interactive legacy systems to Web services[C]. In: Proceedings of the 10th European Conference on Software Maintenance and Reengineering, 2006
4 Reynaud C, Sirot J P, Vodislav D. Semantic integration of XML heterogeneous data sources[C]. In: Proceedings of the 2001 International Symposium on Database Engineering & Applications, 2001. 199～208
5 Abdalla K F. A model for semantic interoperability using XML [C]. In: Proceedings of Systems and Information Engineering Design Symposium, 2003. 107～111
6 Dan A, Johnson R, Arsanjani A. Information as a Service: Modeling and Realization[C]. In: Proceedings of SDSOA, 2007. 2
7 Wang Z, Zhao Z, Fang J. A Service-oriented Approach for Flexible Information Resource Integration [C]. In: Proceedings of COMPSAC, 2007. 573～578
8 Lewis G, Morris E, Smith D, et al. Service-Oriented Migration and Reuse Technique (SMART) [C]. In: Proceedings of 13th IEEE International Workshop on Software Technology and engineering Practice, 2005. 222～229
9 Wang X F, Shawn H, Enamul H, et al. Integrating Legacy Systems within The Service-oriented Architecture[C]. In: Proceedings of Power engineering Society General Meeting, 2007. 1～7
10 Jennex M E. Modeling Emergency Response Systems [C]. In: Proceedings of the 40th Annual Hawaii International Conference on System Sciences, 2007. 22
11 Turoff M, Chumer M, Walle B V, et al. The design of a dynamic emergency response management information system [J]. Journal of Information Technology Theory and Applications, 2004, 5 (4): 1～36
12 Mendonca D, Beroggi G E G, Wallace W A. Decision support for improvisation during emergency response operations [J]. International Journey emergency Management, 2001, 1(1): 30～38