

构建面向服务系统时既有系统可利用性评价研究

车战斌, 韩玉民, 邢颖
(中原工学院 软件学院, 河南 郑州 450007)

摘要: 提出了一种构建面向服务(SOA)的系统, 以及对既有系统的服务提取及可用性评价的方法. 通过对既有系统可提取的服务、与外部系统交换数据是需要的服务之间关系的分析, 提出了既有系统的可利用性评价方法和模型. 然后, 对此方法步骤进行了论证, 通过数学方法提出了一个定量的对既有系统整体可利用性的评价指标. 利用此方法, 可以规范既有系统服务提取的设计过程, 并且能对既有系统的可利用价值进行一定的评判.

关键词: 面向服务的架构(SOA); 既有系统; 可利用性

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2011)08-0187-04

Research on Availability Evaluation of Existing System when Constructing SOA System

CHE Zhan-bin, HAN Yu-min, XING Ying

(College of Software, Zhongyuan Institute of Technology, Zhengzhou 450007, China)

Abstract: This paper propose a method of services extraction and availability evaluation of existing system when constructing a service-oriented (SOA) system. Through analysis of the relationship between the services and exchange data of Existing systems, methods and models of availability evaluation of Existing systems are proposed. Steps of this method have been demonstrated. Through mathematics approach, this paper puts forward a quantitative evaluation index for the whole availability of the existing system. Using this method, you can standardize the design process of existing system services extraction, and it has a certain judge for availability of the existing system.

Key words: service-oriented architecture (SOA); existing system; availability

1 引言

随着企事业单位、政府信息化进程的不断深入, 单位不同时期不同开发者开发的系统间信息不能共享, 不能协同工作的问题越来越严重. 另外, 在新系统开发中也面临对老的系统如何处理的问题. 解决方法有两种, 一个是全部重新设计, 一个是对既有系统进行整合利用. 实际上既有系统很多是经过了很多时间的考验, 能够稳定运行的系统, 也是投入了巨大的人力物力积累下来的单位的宝贵财富, 在新系统开发中保护已有投资, 减少新的投入, 是新的软件开发的重要原则之一^[1].

本文通过对既有系统可提取的服务、与外部系

统交换数据是需要的服务之间关系的分析, 提出了既有系统的可利用性评价方法和模型.

2 面向服务的体系结构

所谓面向服务的体系结构, 就是在软件设计中, 通过“服务”这一功能单位的重组构建新的软件的设计方法.

2.1 服务的定义

面向服务的架构中, 最主要的核心是服务, 对服务的理解决定了 SOA 架构的构建. 从架构的角度来看, 服务就是部署在 SOA 平台上的一个模块, 或者一个功能部分^[2]. 通过使用 SOA 架构, 企业已经投资的信息资源可以通过最小的二次投资来达到融

入到新的大环境、大平台中的效果,可以为各种企业带来较大的利益.关于服务的定义有很多,在本文中定义为满足以下三个条件的软件处理集合:

(1)自我完成性

服务应该不依赖于其他服务的状态,单独执行可能.也就是说不能存在其他处理不进行的话就不能执行,或者输入依赖于其他处理的输出的情况.

(2)开放的接口

服务应该是从外部可以利用的开放的接口,服务的调用方法依赖于所运行的平台,或者仅仅是系统内部的处理等不能成为服务.服务的开发语言或工具可以不同.

(3)粗粒度

服务应该是粗粒度的.就是说单个服务就能够完成一定的业务功能.多个服务的组合可以完成更高层次的、更粗粒度的功能.

这三个条件是构成 SOA 服务的必要条件. SOA 正因为有这个特点,才能对业务过程中的各种变化,通过服务的再组合等方法灵活地重新构造系统.为了下边的讨论方便,我们把包含各种服务的软件集合用下边式子的表示.

$$\text{Feature}(s) = \{f_1, f_1, f_3, \dots, f_n\}$$

2.2 SOA 架构环境特征

SOA 平台是一个抽象的平台,虽然现有的实现主要通过 Web services 技术,但是实际应用中并没有强制要求使用此技术来实现. SOA 架构环境的基本特征如下.

(1)支持多种消息格式和消息模式

SOA 架构中的服务模块来自不同的系统以及不同的平台,这些服务会要求基于不同协议的不同消息之间进行通信.

(2)支持模块之间的互操作

运行状态中的 SOA 平台上的一个服务需要通过消息之间共同协作完成一定的业务功能.要求不同的服务可以通过消息互通完成互操作.

(3)低耦合、高复用性

SOA 架构是一种开放架构,结构上的低耦合对这种架构就非常重要.此方法可以提高系统中服务的重用性,大大提高了企业的信息化投资效率,减少不必要的信息化投资浪费.

2.3 SOA 实现技术标准

当前主流的被接受的实现技术思想是服务组件框架思想.服务组件框架又称为服务组件架构(service component architecture, SCA)规范, SCA

定义了一套可以用来实现面向服务的应用系统或者平台的规范来指导构建编程模型.架构中的服务由定义的服务接口和服务中的实际功能操作两个部分组成.此架构支持多层的的服务调用和服务组合形成新服务的功能.

服务组件实现了一定粒度上的基本业务逻辑,而且作为服务单位被发布出来,外部可以通过 WSDL 来调用服务.服务组件的实现过程中也可以通过规范的 WSDL 接口来调用其他服务,这种调用过程,我们称之为“引用”.基本的服务组件参考模型见图 1.

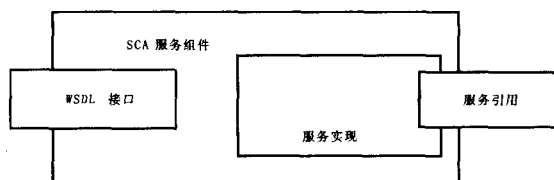


图 1 服务组件模型图

服务组件模块是该平台上最基本的元素,将多个服务组件组合就形成了服务模块.服务模块是 SCA 中引入的第二个概念.通过组合多个服务组件形成一个更大的功能模块来完成功能可以在粒度上合理控制服务组件的大小.控制服务组件的粒度可以增加系统功能模块的重用性.

此外,SCA 还定义了支持多个功能模块之间形成调用关系的导入以及导出功能,通过导入和导出功能可以支持一个服务模块调用其他的服务模块,也让服务模块自己可以被其他的模块调用.最后,服务还定义了共享库来达到最大程度的资源共享.

3 服务的提取方法

3.1 自顶向下的方法

在基于 SOA 的系统设计中,一般都是先对系统的业务流程进行自上而下的分析,然后逐步构件系统.

在自上而下的分析方法中,首先对整个业务层次进行划分,把业务抽象成层次化的功能,对业务进行流程分析,并且用业务流程图表示出来^[3].如图 2 所示.

第二步,把业务流程按照合适的粒度分离成对应的服务.为了能够灵活处理业务环境的变化,合适的服务粒度是非常重要的.这个粒度从业务角度看服务的功能可能的话应该越大越好,并且服务必须是在业务上是独立的.

第三步,设计与服务对应的软件模块.就是要用

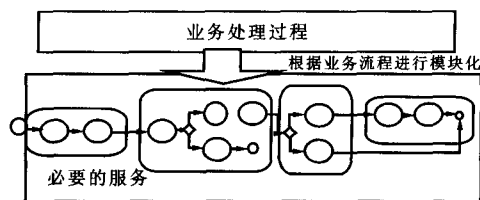


图2 自顶向下的分析流程

合适粒度的模块去实现这些服务。本文中,把按照需要的业务 P ,通过的自上而下的方法提取的必要服务称为 S_n (m 个必要的服务 S_n 的集合表示为

$$S_{nec} = \{S_{n1}, S_{n2}, S_{n3} \dots S_{nm}\}$$

3.2 自底向上的方法

在对现有系统进行服务的抽取时,是从软件代码中分离出符合要求的服务,称为自底向上的方法。具体做法是先把源代码转换成数据流图 (DFD) 然后,根据数据的依赖关系,提取其中相对独立的处理功能作为服务项,如图3所示。具体做法如下:

- 第一步,把源程序转换为 DFD;
- 第二步,对在 DFD 上流通的数据进行分类;
- 第三步,确定数据间的依赖关系;
- 第四步,按照服务抽取的原则进行服务的抽取。

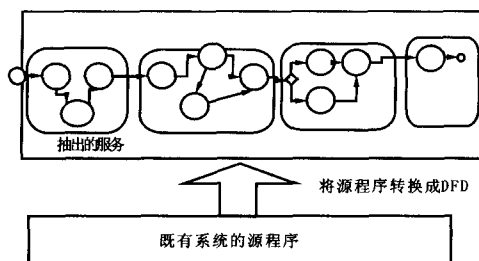


图3 自底向上的分析流程

为了能够提取合理的具有开发性的服务接口,必须认真研究在 DFD 上的数据流,从中分离出开放的数据。再 DFD 上的数据可以分为三类:外部数据、系统数据、模块数据。外部数据就是从系统外部与系统中的处理之间进行交换的数据,在代码上就是系统的文件或者标准输入输出。系统数据是系统内部软件全体公共使用的数据,例如公共变量等。模块数据是指在程序模块中定义的仅在模块内部使用的的数据,例如局部变量。

确定数据依赖关系时可以将依赖关系按照依赖的紧密程度分为模块依赖、处理依赖、系统数据依赖、控制依赖。所谓模块依赖就是数据之间利用输入输出关系产生依赖;所谓处理依赖就是数据之间因为处理的先后顺序而产生的依赖关系;控制依赖指的是处理之间存在一个处理的进行能够决定另一个处理是否能进行。

通过对依赖关系的分析,把那些依赖关系非常密切的功能组合作为服务提取,可以按照以下的原则决定是否将多个处理组合成一个服务:

- (1)将模块数据依赖关系密切的处理单元进行结合;
- (2)将依赖于不同系统数据的处理单元进行分割;
- (3)将处理的过程相互依赖的处理单元进行结合;
- (4)将互相之间通过控制相互依赖的处理单元进行分割。

按照上述原则可以在既有系统中分离出来的服务集合可以表示为

$$S_c = (S_{c1}, S_{c2}, \dots, S_{cm})$$

4 可用性评价方法

4.1 基本思路

前面所谈的自顶向下和自底向上两种方法对于既有系统的利用都是不全面的。前者仅考虑了对既有利用上的需要,后者仅强调了既有系统能够提供什么服务。因此需要对二者进行结合。

按照自顶向下的方法得到的服务,因为没考虑既有系统能够提供的服务情况,所以很难直接能从既有系统中分割出来,按照自底向上分割出来的服务,因为没有考虑系统设计的需要也很难直接用于系统的设计中^[4]。所以,要把分别用以上两种方法得到的服务进行匹配,通过分析其匹配程度再对利用性进行分析。如图4所示。

具体方法可以规范为下列形式化描述。

输入:业务流程 P (待开发系统的业务流程)、既有系统 ES 的源程序 C 。

输出: ES 的可利用性结论。

处理:

Step1,服务提取。对业务流程 P 进行自顶向下的分析,对既有系统的源程序 C 进行自底向上的分析,分别得到 S_n 和 S_c ;

Step2,服务的再利用性进行判定。就是将 S_c 的各个功能与 S_n 的各个服务进行比较,判断各个候选的 S_c 的可用性;

Step3,对 ES 的可再利用性进行量化分析。

4.2 服务提取

首先,对业务系统进行自顶向下分析,从业务的角度提取对适当粒度的服务集合 $S_{nec}(p)$ 。

然后,对既有系统的源程序进行自底向上的分

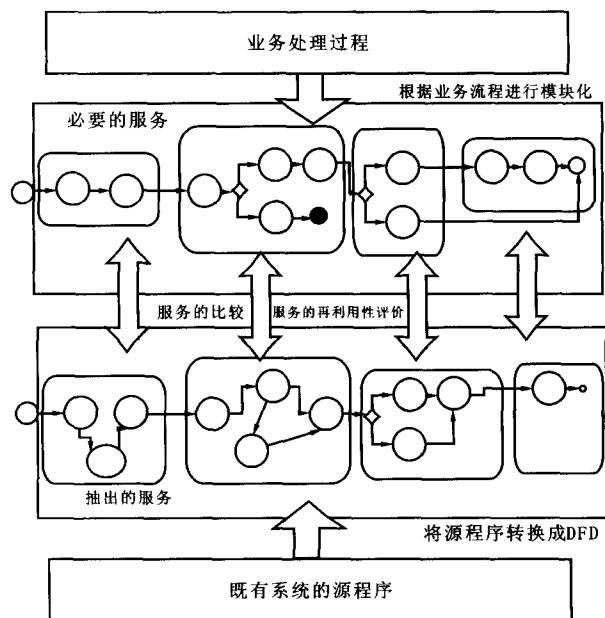


图4 综合方法分析流程

析,得到既有系统候选的可利用服务集合 $S_{\text{cand}}(c)$. 这时,应该先对源程序进行尽可能细粒度的服务提取,然后再通过组合,提取出合适粒度的服务.

4.3 服务可利用性的判别

这里所说的服务的可利用性是指,针对 $S_{\text{nec}}(p)$ 中各个必要的服务, $S_{\text{cand}}(c)$ 中的候选服务在多大程度上是可利用的. 因此对服务的可利用性定义如下:

定义1 设已经存在必要服务 $s_n \in S_{\text{nec}}(p)$ 和候选服务组合 $s_c \in S_{\text{cand}}(c)$. 对于 s_n 来说, s_c 的可利用性定义为 $\text{feature}(s_n) \supseteq \text{feature}(s_c)$ 成立. 称 s_c 为 s_n 的可利用服务. s_n 的全部可利用服务的集合记为 $S_{\text{re}}(s_n)$. 这里, $S_{\text{re}}(s_n) \subseteq S_{\text{cand}}(c)$. 显然,这里的候选服务 s_c 全部功能肯定少于 s_n . s_n 最终要由既有系统的 s_c 和其他的候选服务或者新开发的服务共同实现.

4.4 可利用性的定量描述

为了定量描述既有系统的可利用性,可以用必要服务 s_n 中候选服务 s_c 可以提供的服务比例来描述既有系统的可利用性. 功能的复杂性,在软件工程中有很多量化的描述方法,但是一般来说在评估既有系统可用性阶段,待开发系统的代码行数一般是未知的,另外,由于开发语言的不同,既有系统的代码行数也不具备可比性. 我们可以采用功能点法(FP)来计量功能的复杂度.

对于功能 f 的功能点数可以记为 $C_{\text{fp}}(f)$, 这样,服务 s 的功能点数可以记为

$$C_{\text{fp}}(s) = \sum_{f \in \text{feature}(s)} C_{\text{fp}}(f)$$

式中, $f \in \text{feature}(s)$, 表示 f 是服务 s 中的全部

功能.

服务的可利用率定义如下所示.

定义2 设必要服务为 $s_n \in S_{\text{nec}}(p)$, 对于必要服务 s_n 的候选服务的可利用率定义为

$$P_{\text{re}}(s_n) = \frac{C_{\text{fp}}(\bigcup_{s_{\text{ci}} \in S_{\text{re}}(s_n)} s_{\text{ci}})}{C_{\text{fp}}(s_n)}$$

其中分子部分表示实现 s_n 的所有既有系统候选服务集合的总功能点数. 分母表示必要服务 s_n 的总功能点数. 显然 $P_{\text{re}}(s_n)$ 小于 1. $P_{\text{re}}(s_n)$ 越大说明既有系统中能够可用来完成服务 s_n 的既有服务越多.

这样,既有系统总体的可利用性就可以表示为所有 $P_{\text{re}}(s_n)$ 的加权和,这里的加权是该必要功能 s_n 的功能点数占总体系统功能点的比例. 即:

$$\begin{aligned} P_{\text{re}}(\text{ES}) &= \sum_{s_n \in S_{\text{nec}}} \frac{C_{\text{fp}}(s_n)}{\sum_{s_n \in S_{\text{nec}}} C_{\text{fp}}(s_n)} P_{\text{re}}(s_n) \\ &= \frac{\sum_{s_n \in S_{\text{nec}}} C_{\text{fp}}(S_{\text{re}}(s_n))}{\sum_{s_n \in S_{\text{nec}}} C_{\text{fp}}(s_n)} \end{aligned}$$

这里, ES 表示既有系统. 这个式子表明,既有系统总体可利用性可以用既有系统可利用的功能点数与待开发系统的总体功能点数的比表示.

需要说明的是:

(1) 这里的 S_{nec} 并非待开发系统的全部功能而是与既有系统有关,希望在既有系统中得到的功能. 通常,待开发系统都是几个既有系统的综合.

(2) $S_{\text{re}}(s_n)$ 是表示为了实现服务 s_n 在既有系统中可利用的服务. 通常由于既有系统的自治性功能很多,所以这些功能仅仅是既有系统的一小部分.

5 结束语

本文提出了在构建新的 SOA 系统时既有系统的服务提取方法及既有系统进行可用性评价的方法和模型. 利用此方法,可以规范既有系统服务提取的设计过程,并且能对既有系统的可利用价值进行一定的评判. 本方法已经在一些项目的开发中得到部分使用和验证. 结果证明,对于开发新的基于 SOA 的系统的既有系统评价和成本预测具有一定价值.

本文的研究还处在开始阶段,今后要在以下几个方面继续开展工作.

(1) 继续研究既有系统服务提取时如何更合理的进行服务的合并、分割,提出更规范化的办法.

(下转第 194 页)

求转移到了恶意系统管理员控制的节点,否则认为节点是可信赖的. TC 解密会话密钥,并在消息 3 中发送给节点 N ,此时 N 就可以解密并启动 VM 了. 最后节点发送消息 4 给用户,消息包含节点运行 VM 的证明.

在实时迁移^[6]中,运行中的 VM 的状态信息在两个节点间迁移:源节点 N_s 和目标节点 N_d . 为保证操作的安全性,两个节点必须互信,且 VM 状态必须可信并且在完成迁移前是不可修改的. 图 5 所示为参与 VM 安全迁移的消息队列,首先 N_s 请求 TC 检验 N_d 是否可信,消息 3 中, N_s 向 N_d 发起 VM 迁移申请并附加会话密钥 K_s . N_d 验证 N_s 是否可信(第 4、5 步). 如果两个节点都认证成功, N_d 通知 N_s 接受 K_s (第 6 步),在消息 7 中 N_s 最终将 VM 状态哈希值加密发送给 N_d ,VM 迁移成功.

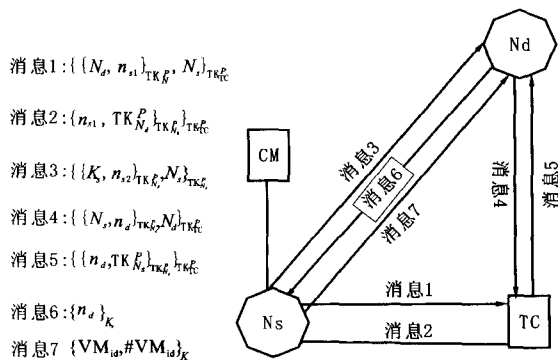


图 5 VM 迁移过程中的消息交换

4 结束语

企业普遍应用云计算的主要阻力源自对数据和计算的保密性和完整性的担心,本文提出一种可信云计算平台 TCCP,它可以为 IaaS 服务提供一个封闭执行环境,保证了客户 VM 执行的机密性,允许

用户验证 IaaS 提供商并在装载其虚拟机前先判断服务是否安全.

参考文献:

- [1] Berger S, Caceres R, Goldman K A. vTPM: virtualizing the trusted platform module[C]// Proc of USENIX-SS'06, Berkeley, CA, USA:IEEE, 2006.
- [2] Survey: cloud computing no hype, but dear of security and control slowing adoption[EB/OL]. [2010-10-15]. http://www.circleid.com/posts/20090226_cloud_computing_hype_security/.
- [3] Clark C, Fraser K, Hand S. Live migration of virtual machines[C]// Proc of NSDI'05. Berkeley, CA, USA: IEEE, 2005:273-286.
- [4] Murray D G, Milos G. Improving xen security through disaggregation[C]// Proc of VEE'08. New York, NY, USA:IEEE, 2008.
- [5] Nurmi D, Wolski R, Grzegorzczuk C. Eucalyptus: a technical report on an elastic utility computing architecture linking your programs to useful systems. Technical Report 2008-10[R]. Santa Barbara California UCSB Computer Science, 2008.
- [6] TCG. [EB/OL]. [2010-10-15]. <https://www.trustedcomputinggroup.org>.

作者简介:

沈松男,(1973-),副教授.研究方向为网络安全和云计算.

邢剑锋男,(1981-),硕士,讲师.研究方向为 P2P 和云计算.

王鹏飞男,(1984-),硕士,讲师.研究方向为网络安全和光纤通信.

朱飞男,(1979-),讲师.研究方向为网络安全和云计算.

(上接第 190 页)

(2)对文中的办法用更多的实际系统进行验证,发现问题,改进方法.

参考文献:

- [1] 木村隆洋,中村匡秀,井垣宏,松本健一,“データ依存解析に基づくレガシーソフトウェアからのサービス提取法,”信学技報 ソフトウェアサイエンス研究会[J]. 2005(45):13-18.
- [2] Papazoglou M P, Georgakopoulos D. Service Oriented Computing[J]. Communications of the ACM, 2003,46(10):25-28.

[3] 牧野友紀. ビジネス環境と実装システムを繋ぐ BPM と SOA[J]. 情報処理, 2005,46(1):60-63.

[4] 田中 秀一郎. サービス指向に基づくビジネスプロセス分析方法論[D]. 日本:日本南山大学,2006.

作者简介:

车战斌男,(1964-),博士,副教授.研究方向为分布式系统、分布式数据库.

韩玉民男,(1963-),本科,高级工程师.研究方向为软件开发技术、软件技术教育.

邢颖女,(1985-),硕士,助教.研究方向为网格计算与云计算.