维普资讯 http://www.cqvip.com

C/S 遗留系统到 SOA 系统移植框架研究*)

谢 刚[1,2] 张为群[1]

(西南大学计算机科学与信息学院软件工程研究室 重庆 4000715)[1]

(贵州师范大学数学与计算机科学学院 贵阳 550001)[2]

摘 要 遗留系统在组织中起着重要的作用。随着商业环境的快速变化,应该不断用新技术对其进行进化。Web Service 和面向服务的体系结构(SOA)使我们能在面向服务的计算环境中进化遗留系统。用 Web Service 技术集成遗留系统的传统方法是将遗留系统包装成"黑盒子"。本文提出了一种再工程方法,该方法将遗留系统的包装和理解相结合。该方法不仅能对服务进行识别和包装,还可以将遗留系统移植到 SOA 的体系结构中。

关键词 C/S,SOA,遗留系统,移植

A Framework for Migrating C/S Architectual Legacy Systems to the SOA

XIE Gang[1,2] ZHANG Wei-Qun[1]

(Software Engineering Laboratory of College of Computer Science, Southwest University, Chongqing 4000715)[1]

(College of Mathematics and Computer Science, Guizhou Normal University, Guiyang 550001)[2]

Abstract Legacy system is vital to organizations. With the increasing change of business environment, they are evolved with new emerged technologies. Web Service and service-oriented architectures enable us to evolve legacy system in service-oriented computing environment. Traditional approaches to integrate legacy systems with Web service technology are wrapping a legacy system as a black-box without adequate system understanding. In this paper, we propose a reengineering approach which package and understanding the legacy systems. It not only can identify and package service, but also migrate legacy system into service-oriented architectures.

Keywords C/S,SOA,Legacy system,Migrating

1 引言

遗留系统在组织中起着重要的作用。随着商业环境的快速变化,应该不断用新技术对其进行进化。Web Service 和面向服务的体系结构(SOA)使我们能在面向服务的计算环境中进化遗留系统。

目前,将遗留系统进化为 SOA 有三种方法[1]:

①将遗留系统包装成 Web Service。利用这种方法,只需分析遗留接口,不需知道遗留系统的内部细节。从长远来看,该方法使遗留系统的维护和管理更加复杂。

②利用业务逻辑将遗留系统再工程为 Web Service。这种方法利用逆向工程技术从源代码中恢复业务逻辑,并根据恢复的业务逻辑开发一个新的系统。在实现 Web Service 的同时利用了有价值的业务逻辑和新技术。从而节约了维护费用,提高遗留代码事务处理的效率。然而,完整地恢复业务逻辑是一件非常困难的事。

③将对遗留系统的包装和理解相结合。这种方法又称为灰盒方法。该方法以需求分析和系统再工程为基础,能快速可靠地集成系统和有用的业务逻辑。该方法解决了工业和学术在面向服务计算环境中进化遗留系统的有关分歧。

本文基于这个想法描述了一种遗留系统移植方法。

2 相关定义

定义 2.1(遗留系统移植的定义) $N=f(O)$,其中 O 表示 C/S 的遗留系统,N 表示 SOA 新系统,f 表示移植方法。

定义 2.2(遗留系统的定义) $O=\{G_1,G_2,\cdots,G_n\}$,其中 $G_i(i=1,2,\cdots,n)$表示遗留系统中的第 i 个子系统。

定义 2.3(G_i 的定义) $G_i=\{S_1,S_2,\cdots,S_n\}$其中 $S_j(j=1,2\cdots,n)$表示第 i 个子系统中的第 j 个函数(子程序或子过程)

定义 2.4(S_j 的定义) $S_j=\{C_1,C_2,\cdots,C_n\}$其中 $C_i(i=1,2,\cdots,n)$表示第 j 个函数(子程序或子过程)中的第 i 个类。

定义 2.5(*SOA* 新系统的定义) $N=\{A_1,A_2,\cdots,A_n\}$,其中 $A_k(K=1,2,\cdots,n)$表示第 k 个服务。

定义 2.6(问题域的定义) $P=\{P_1,P_2,\cdots,P_n\}$,其中 $P_i(i=1,2,\cdots,n)$表示问题域中的第 i 个问题。

3 移植框架中的关键问题

大型遗留系统往往有丰富的、复杂的结构。然而,它可被分割成相对独立的子系统。当尽可能地将接口进行封装,就可达到松耦合。如果一个软件实体是独立的、自我包含的、粗粒度的、松耦合的,它将成为软件服务的候选对象。在这部分,将分步骤地描述我们的再工程方法。该方法解决了从服务识别到服务集成的相关问题。图 1 给出了移植的框架。

3.1 评估遗留系统

对遗留系统进行评估,评估是为了反映遗留系统的当前状态和说明遗留系统属于生命周期的哪个阶段。为了对多种因素进行综合考虑,在评估中使用决策树。根据评估结果,决定再工程决策。如果遗留系统满足以下因素,那么将它移植

*)基金项目:重庆市科委重庆市自然科学基金(CSTC,2006BA2003)。谢 刚 助教,硕士研究生;通讯作者:张为群 教授,硕士生导师。

维普资讯 http://www.cqvip.com

到 SOA 中是合适的。

①业务逻辑中包含可重用的且可靠的功能。

②与维护整个系统相比，遗留系统中的可重用构件更易维护。

③从需求的角度来说，能够将功能作成一个独立的服务。

④客户端是能分解的。

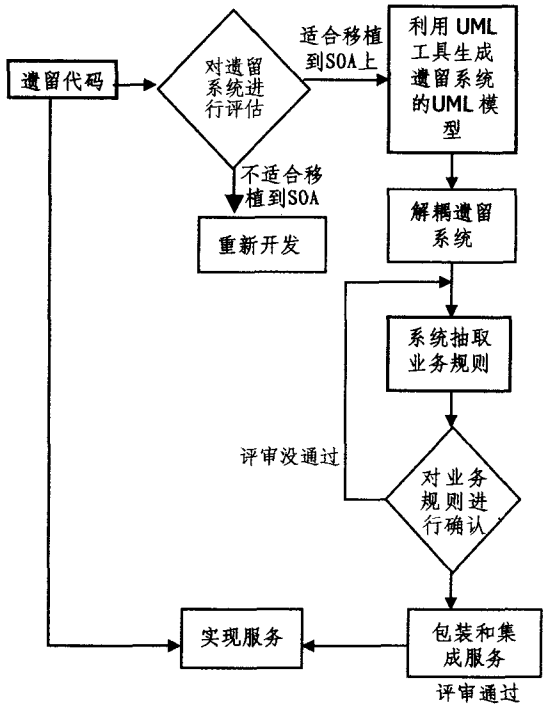

图 1　移植过程

3.2　解耦遗留系统

由于长时间的进化，使系统具有高耦合。文档不能反映遗留系统的状态。许多情况下，需要将遗留系统分解成低耦合，高内聚的系统[2]。软件聚集技术被用来捕捉可重用的、独立的、自我包含的、精粒度和松耦合的遗留代码段[8]。

聚类分析根据实体的关系和相似度将数据集中的实体进行聚类。它被用于程序理解领域。在这里，我们应用聚类算法来抽取遗留代码中的独立的服务。这个聚类方法分析遗留代码并用树状图表示结果，该结果表示遗留系统的层次。

通过聚类分析和人的干预就可以得到粗粒度和松耦合的构件。

3.3　聚类分析

我们在理解源代码的过程中应用聚类技术。为此，我们对聚类算法的功能进行了改进。

3.3.1　相关定义

定义 3.1(实体集的定义)　$E=\{E_1,E_2,\cdots,E_n\}$其中 E_i $(i=1,2,\cdots,n)\in G_i$ 或 $E_i\in S_j$。

定义 3.2(特征集的定义)　$F_i=\{F_{i1},F_{i2},\cdots,F_{in}\}$其中 F_{ij} $(j=1,2,\cdots,n)$表示第 i 个实体的第 j 个特征，F_i 表示第 i 个实体的特征集。F_{ij} 的值为第 i 个实体中每个标识符出现的概率。

定义 3.3(特征的权重 w_f)　赋于每个特征的权值，$w_f>0$。与目标服务有关的标识符的权重较高。

定义 3.4[3](特征集的权重)

$$W(F_i)=\sum_{f\in F_i} w_f$$

定义 3.5[3](实体间的联系)

$$Linked(E_i,S)=\sum_{E_j\in S} Linked(E_i,E_j)$$

(如果 E_i 调用 E_j 或 E_i 被 E_j 继承或 E_j 调用 E_i 或 E_j 被 E_i 继承，那么 $Linked(E_i,E_j)=1$；否则 $Linked(E_i,E_j)=0$)

定义 3.6[3](实体相似度的计算)

$$\mathrm{Sim}(E_i,E_j)=\frac{W(F_i\cap F_j)+Linked(S,E_i)+Linked(S,E_j)}{W(F_i\cap F_j)+W(F_i-F_j)+W(F_j-F_i)}$$

定义 3.7[3]　假设当前所处的聚集层为 n，那么称第 n 层产生的聚集实体为新聚集实体，第 n 层未聚集的实体为旧聚集实体。

定义 3.8[3]　实体与新聚集实体之间的相似度定义如下：

$$\mathrm{Sim}[(k),(a,b)]=\mathrm{MAX}\{\mathrm{Sim}[(k),(a)],\mathrm{Sim}[(k),(b)]\}$$

定义 3.9(相似度矩阵的定义)　$D_{n*n}=[\mathrm{Sim}(i,j)]$其中 n 表示实体数，$i=1,2,\cdots,n,j=1,2,\cdots,n$。

为了方便叙述，定义下列基本符号：

①$C(h)$表示聚集的层次为 h；

②(E_i)表示对实体 E_i 的聚集。

3.3.2　实现算法

①令 $C(0)=0,n=0$；

②识别出 O 中的所有 E_i；

③计算相似度矩阵 $D=\{\mathrm{Sim}[(E_i),(E_j)]\}$；

④求 $\max(D)$所对应的聚集实体对$(E_i),(E_j)$；

⑤$n=n+1$；

⑥$(E_n)=((E_i),(E_j))$

⑦$C(n)=\mathrm{Sim}((E_i),(E_j))$；

⑧删除聚集(E_i)所对应的行和列，(E_j)所对应的行和列。并为(E_n)增加一行与一列。

⑨计算新聚集(E_n)与老聚集(k)的相似度

⑩相似度矩阵只有一个元素时，算法停止。否则，转到④。

3.4　业务规则抽取

尽管得到独立的和松耦合的构件，但在构件中存在很多无关的代码，这时就需要把核心的业务功能代码段抽取出来，这些代码段的组合就是业务规则，形式化定义为 O=S(I)其中 I 为输入变量集合，O 为输出变量集合，S 为 I 变为 O 所要执行的代码段。

业务规则抽取的步骤如下：

Step1：识别出每个构件中的输入变量。

Step2：利用程序切片技术[4]得到输入变量有关的代码段，记作 S1

Step3：识别出每个构件功能相关的输出变量。

Step4：利用程序切片技术得到输出变量有关的代码段，记作 S2。

Step5：比较 S1 和 S2，找出与输入变量和输出变量都有关的代码段即为我们所求的业务规则。

3.5　业务规则确认

对所抽取的业务规则要进行确认，确认的标准如下：

①公平的表示：从代码中抽取的规则必须要反映软件的状态。

②一致性：是否与该软件的域模型一致。

3.6　服务包装和集成

服务包装是将候选的遗留服务变成功能性服务的必要步骤。因为一个面向服务的体系结构鼓励单个服务自我包含。

维普资讯 http://www.cqvip.com

第一：对“胶水”代码进行包装。

第二：以抽取的业务规则为基础，设计服务接口。使用WSDL对服务的相关数据进行描述。如果目标服务是一个Web Service，那么就用SOAP处理器进行集成。

第三：创建Web Service后，它将在UDDI中进行注册。UDDI能使公司和应用快速地、容易地和动态地发现和使用因特网上的Web服务。

第四，在面向服务体系结构中通过服务组合来改进遗留系统。为了在面向服务的体系结构中建立更强的合作服务[5~7]，需将遗留代码中的服务与其他Web服务或基于服务的应用组成起来。最终的合作服务将更有价值。

4 实例

我们以一个高等教育再工程项目为例来说明移植过程中所遇到的一些问题。在一个大学的计算机学院提供计算机职业培训程序。随着计算机职业培训需求的增长，计算机学院决定向国际学生提供该培训。管理计算机职业培训课程的软件系统被使用已经维护了三年。为了适应所有的新的计算机职业培训程序的新需求，计算机学院希望将基于局域网的遗留系统移植到面向服务的体系结构中。

遗留计算机职业培训管理系统是C/S结构的。客户端用C++开发的。所有的业务逻辑在客户端被实现。它是一个典型的胖客户应用。主要配置WINDOWS NT和一个桌面数据库。这个信息系统已被修改了很多次。代码中的许多部分都无完整的文档，再工程这样的系统是一个极大的挑战。

4.1 抽取遗留代码

我们开发了一个原型工具来辅助移植，该工具称为SOAT。

Step1：SOAT对注册服务程序进行语法分析，并检测到它是由60个类和1000个成员和非成员函数组成(MFC类除外)，同时检测出有1200个C++程序实体和它们之间的2400种依赖关系。

Step2：SOAT创建一个UML类图来表示这个遗留系统。

Step3：SOAT生成以聚集算法为基础的树形图。根据遗留系统课程管理模型和新的服务模型来确定接入点。

Step4：抽取业务规则。

Step5：利用JAVA胶水代码对C++遗留代码进行包装并与Axis soap处理器进行绑定。

4.2 移植后的面向服务体系结构

图2为移植后的体系结构。表示重要业务逻辑的C++遗留代码是注册服务进化的核心。在大学中的局域网，客户被进化为Web Service客户，它容易被维护。这个计算机培训注册Web Service已经在公共的UDDI进行注册且WSDL绑定信息被公布。这使已被注册的Web Service能集成所有不同平台的请求者。例如：全球的计算机培训机构在UDDI注册中进行搜索，并发现计算机学院提供一种高质量的计算机职业培训程序。根据WSDL提供的文档，培训机构的Web Service能动态地与计算机学院的服务进行绑定。另外，利用高等教育Web端口来集成已注册的Web Service。这个端口是以.NET为基础的。即在面向服务的体系结构中大量集成已注册的Web Service。

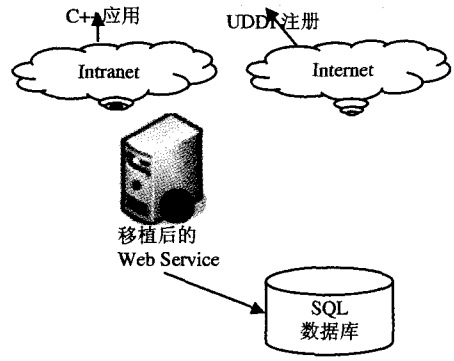

图2 移植后的体系结构

结论和将来的工作 与相似的研究相比，这种方法从服务的角度来再工程遗留系统。它包程序转换、重构、包装和建模等技术。该方法对系统的理解更灵活，使新的系统更容易维护，并降低维护费用。我们主要利用一个聚集方法来分解遗留系统。

在将来，我们将减少人的干预，并利用业务规则的确认机制进行深入的研究，使之能更加客观对其进行评价。

参考文献

1 Zhang Z, Yang H. One-Stone-Two-Birds: Legacy System Re-engineering and Web Services Development——A Component-Based and Service-Oriented Approach. In: Proceedings of Postgraduate Research Conference in Electronics, Photonics, Communications & Networks, and Computing Science (PREP 2004), Hatfield, UK, 2004

2 Wiggerts T A. Using Clustering Algorithms in Legacy Systems Remodularization. In: Proceedings of the 4th Working Conference on Reverse Engineering (WCRE '97), Amsterdam, Netherlands, 1997

3 Zhang Zhoupeng, Yang. Hongji. Incubating Services in Legacy Systems for Architectural Migration. IEEE

4 Tip F. A Survey of Program Slicing Techniques[J]. Journal of Programming Languages, 1995, 3(1): 121~189

5 Chang H, et al. QoS-Aware Middleware for Web Services Composition. IEEE Transactions on Software Engineering, 2004, 30(5): 311~327

6 Fremantle P, Weerawarana S, Khalaf R. Enterprise Services. Communications of the ACM (CACM), 2003, 46(10): 35~40

7 Yang J. Web Service Componentization: Towards Service Reuse and Specialization. Communications of the ACM (CACM), 2003, 46(10): 35~40

8 Li Jian-zhi, Zhang Zhuo-peng, Qiao Bing, et al. Component Mining Approach to Incubate Grid Services in Object-Oriented Legacy Systems. International Journal of Automation and Computing, 2006(1): 47~55